



Guia do Desenvolvedor

# Amazon Simple Notification Service



# Amazon Simple Notification Service: Guia do Desenvolvedor

Copyright © 2024 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

As marcas comerciais e imagens de marcas da Amazon não podem ser usadas no contexto de nenhum produto ou serviço que não seja da Amazon, nem de qualquer maneira que possa gerar confusão entre os clientes ou que deprecie ou desprestige a Amazon. Todas as outras marcas comerciais que não são propriedade da Amazon pertencem aos respectivos proprietários, os quais podem ou não ser afiliados, estar conectados ou ser patrocinados pela Amazon.

---

# Table of Contents

O que é o Amazon SNS? .....	1
Recursos e capacidades .....	3
Serviços relacionados .....	5
Acessando o Amazon SNS .....	5
Preços do Amazon SNS .....	6
Cenários comuns do Amazon SNS .....	6
Integração de aplicações .....	6
Alertas do .....	7
Notificações ao usuário .....	7
Notificações por push para dispositivos móveis .....	8
Trabalhando com AWS SDKs .....	8
Origens e destinos de eventos do Amazon SNS .....	10
Origens de eventos .....	10
Análises .....	10
Integração de aplicações .....	11
Faturamento e gerenciamento de custos .....	11
Aplicações de negócios .....	12
Computação .....	12
Contêineres .....	14
Envolvimento do cliente .....	14
Banco de dados .....	15
Ferramentas de desenvolvedor .....	16
Web e móveis de front-end .....	17
Desenvolvimento de jogos .....	18
Internet das Coisas .....	18
Machine learning .....	19
Gerenciamento e governança .....	20
Mídia .....	22
Migração e transferência .....	23
Redes e entrega de conteúdo .....	24
Segurança, identidade e conformidade .....	25
Sem servidor .....	26
Armazenamento .....	27
Fontes de eventos adicionais .....	29

Destinos de eventos .....	30
Destinos A2A .....	30
Destinos A2P .....	31
Configuração .....	34
Criar uma conta e um usuário do IAM .....	34
Inscreva-se para um Conta da AWS .....	34
Criar um usuário com acesso administrativo .....	35
Próximas etapas .....	36
Conceitos básicos .....	37
Pré-requisitos .....	37
Etapa 1: Criar um tópico .....	37
Etapa 2: criar uma assinatura para o tópico .....	37
Etapa 3: Publicar uma mensagem no tópico .....	38
Etapa 4: Excluir a assinatura e o tópico .....	39
Próximas etapas .....	39
Configurar o Amazon SNS .....	41
Criar um tópico .....	41
AWS Management Console .....	42
AWS SDKs .....	45
Inscrever o em um tópico .....	59
Como inscrever um endpoint em um tópico do Amazon SNS .....	59
Excluir uma assinatura e um tópico .....	60
AWS Management Console .....	61
AWS SDKs .....	61
Marcação .....	71
Marcação para alocação de custos .....	71
Marcação para controle de acesso .....	72
Marcação para pesquisa e filtragem de recursos .....	73
Configurar tags .....	74
Ordenação de mensagens e desduplicação (tópicos FIFO) .....	81
Caso de uso de tópicos FIFO .....	81
Detalhes de ordenação de mensagens .....	83
Agrupamento de mensagens .....	86
Distribuição de dados por IDs de grupo de mensagens para melhor desempenho .....	87
Entrega de mensagens .....	88
Filtragem de mensagens .....	89

Desduplicação de mensagens .....	90
Segurança de mensagens .....	93
Durabilidade das mensagens .....	94
Arquivamento e reprodução de mensagens .....	96
O que é arquivamento e reprodução de mensagens? .....	96
Para proprietários de tópicos .....	97
Para assinantes do tópico .....	102
Exemplos de código .....	106
Exemplo FIFO (AWS SDKs) .....	106
Exemplo de FIFO (AWS CloudFormation) .....	119
Publicação de mensagens .....	124
AWS Management Console .....	124
AWS SDKs .....	126
Grandes cargas úteis de mensagens .....	149
Biblioteca do cliente em versão ampliada para Java .....	149
Biblioteca do cliente em versão ampliada para Python .....	154
Atributos de mensagens .....	157
Validação e itens de atributos de mensagem .....	158
Tipos de dados .....	159
Atributos de mensagens reservados para notificações móveis por push .....	160
Agrupamento de mensagens em lotes .....	162
O que é o agrupamento de mensagens em lotes? .....	162
Como funciona o agrupamento de mensagens em lotes? .....	162
Exemplos .....	163
Filtragem de mensagens .....	167
Escopo de política de filtro de assinaturas .....	167
Políticas de filtro de assinatura .....	168
Exemplo de políticas de filtro .....	169
Restrições da política de filtro .....	172
Lógica E/OU .....	176
Correspondência de chaves .....	181
Correspondência de valores numéricos .....	183
Correspondência de valores de string .....	186
Aplicar uma política de filtro de assinatura .....	193
AWS Management Console .....	193
AWS CLI .....	194

AWS SDKs .....	195
API do Amazon SNS .....	199
AWS CloudFormation .....	200
Remover uma política de filtro de assinatura .....	200
AWS Management Console .....	200
AWS CLI .....	201
API do Amazon SNS .....	201
Proteção de dados de mensagens .....	202
O que é proteção de dados de mensagens .....	202
Por que usar a proteção de dados de mensagens? .....	203
Políticas de proteção de dados .....	203
O que são políticas de proteção de dados? .....	204
Visão geral da estrutura das políticas de proteção de dados .....	204
Como faço para determinar as entidades principais do IAM? .....	207
Operações de política de proteção de dados .....	208
Exemplos de política de proteção de dados .....	216
Criar políticas de proteção de dados .....	223
Criar políticas de proteção de dados .....	233
Identificadores de dados .....	234
Identificadores de dados gerenciados .....	234
Identificadores de dados personalizados .....	275
Entrega de mensagens .....	278
Entrega de mensagens brutas .....	278
Habilitar a entrega de mensagens brutas usando o AWS Management Console .....	279
Exemplos de formatos de mensagens .....	279
Atributos de mensagem e entrega de mensagens brutas para assinaturas do Amazon SQS .....	280
Entrega entre contas .....	280
Proprietário da fila cria a assinatura .....	281
Um usuário que não é proprietário da fila cria uma assinatura .....	283
Como faço para forçar uma assinatura a exigir autenticação em solicitações de cancelamento de assinatura? .....	286
Entrega entre regiões .....	286
Regiões de adesão .....	286
Status de entrega de mensagem .....	289
Configurar registro em log do status de entrega usando o AWS Management Console .....	290

Configurando o registro do status de entrega usando os SDKs AWS .....	291
AWS Exemplos de SDK para configurar atributos de tópicos .....	293
Configurar registro em log do status de entrega usando o AWS CloudFormation .....	302
Novas tentativas de entrega de mensagens .....	303
Protocolos e políticas de entrega .....	303
Estágios da política de entrega .....	304
Como criar uma política de entrega HTTP/S .....	305
Filas de mensagens mortas (DLQs) .....	311
Por que há falha nas entregas de mensagens? .....	312
Como as filas de mensagens mortas funcionam? .....	313
Como as mensagens são movidas para uma fila de mensagens mortas? .....	313
Como posso mover mensagens de uma fila de mensagens mortas? .....	314
Como posso monitorar e registrar em log filas de mensagens mortas? .....	314
Configurar uma fila de mensagens mortas .....	315
Arquivamento e análise de mensagens .....	320
Sistema de mensagens de aplicação para aplicação (A2A) .....	321
Fanout para streams de entrega do Firehose .....	321
Pré-requisitos .....	322
Inscrever um fluxo de entrega em um tópico .....	324
Destinos de fluxos de entrega .....	325
Exemplo de caso de uso .....	339
Fanout para funções do Lambda .....	351
Prerequisites .....	351
Inscrever uma função em um tópico .....	352
Fanout para filas do Amazon SQS .....	353
Inscrever uma fila em um tópico .....	354
Exemplo (AWS CloudFormation) .....	361
Fanout para endpoints HTTP(S) .....	369
Inscrever um endpoint em um tópico .....	371
Verificar assinaturas de mensagens .....	379
Analisar formatos de mensagens .....	383
Fanout para AWS Event Fork Pipelines .....	393
Como o AWS Event Fork Pipelines funciona .....	394
Implantar o AWS Event Fork Pipelines .....	398
Implantar e testar o AWS Event Fork Pipelines .....	399
Inscrever um pipeline de eventos em um tópico .....	410

Usar o Agendador do EventBridge .....	419
Configurar o perfil de execução .....	420
Criar uma programação .....	420
Recursos relacionados .....	425
Sistema de mensagens de aplicação para pessoa (A2P) .....	426
Mensagens de texto em dispositivos móveis (SMS) .....	426
Sandbox de SMS .....	427
Identidades de origem .....	432
Solicitar suporte para SMS .....	519
Definir preferências de SMS .....	536
Enviar mensagens SMS .....	543
Monitorar e atividade de SMS .....	566
Gerenciar assinaturas de SMS .....	576
Países e regiões compatíveis .....	607
Práticas recomendadas para SMS .....	628
Notificações por push para dispositivos móveis .....	645
Como funcionam as notificações ao usuário .....	646
Visão geral do processo de notificação ao usuário .....	646
Configurar uma aplicação móvel .....	647
Enviar notificações por push para dispositivos móveis .....	667
Atributos de aplicativo móvel .....	682
Eventos de aplicativo móvel .....	686
Ações da API de push para dispositivos móveis .....	689
Erros de API de push para dispositivos móveis .....	691
TTL de push para dispositivos móveis .....	703
Supported Regions (Regiões compatíveis) .....	706
Práticas recomendadas para notificações por push móveis .....	707
Notificações por e-mail .....	708
AWS Management Console .....	708
AWS SDKs .....	709
Exemplos de código .....	740
Ações .....	750
CheckIfPhoneNumberIsOptedOut .....	751
ConfirmSubscription .....	758
CreateTopic .....	764
DeleteTopic .....	778

GetSMSAttributes .....	788
GetTopicAttributes .....	794
ListPhoneNumbersOptedOut .....	805
ListSubscriptions .....	808
ListTopics .....	821
Publish .....	834
SetSMSAttributes .....	857
SetSubscriptionAttributes .....	862
SetSubscriptionAttributesRedrivePolicy .....	867
SetTopicAttributes .....	868
Subscribe .....	876
TagResource .....	906
Unsubscribe .....	910
Cenários .....	919
Criar um endpoint de plataforma para notificações por push .....	919
Criar e publicar em um tópico FIFO .....	922
Publicar mensagens SMS em um tópico .....	934
Publicar uma mensagem grande .....	940
Publicar uma mensagem de texto SMS .....	944
Publicar mensagens em filas .....	952
Exemplos sem servidor .....	1015
Invocar uma função do Lambda em um acionador do Amazon SNS .....	1016
Exemplos entre serviços .....	1025
Criar uma aplicação para enviar dados para uma tabela do DynamoDB .....	1026
Criação de uma aplicação do Amazon SNS .....	1027
Criar uma aplicação com tecnologia sem servidor para gerenciar fotos .....	1029
Criar uma aplicação de exploração do Amazon Textract .....	1033
Detectar pessoas e objetos em um vídeo .....	1035
Publicar mensagens em filas .....	1035
Usar o API Gateway para invocar uma função do Lambda .....	1036
Usar eventos programados para invocar uma função do Lambda .....	1038
Segurança .....	1040
Proteção de dados .....	1040
Criptografia de dados .....	1042
Privacidade do tráfego entre redes .....	1060
Segurança de proteção de dados de mensagens .....	1077

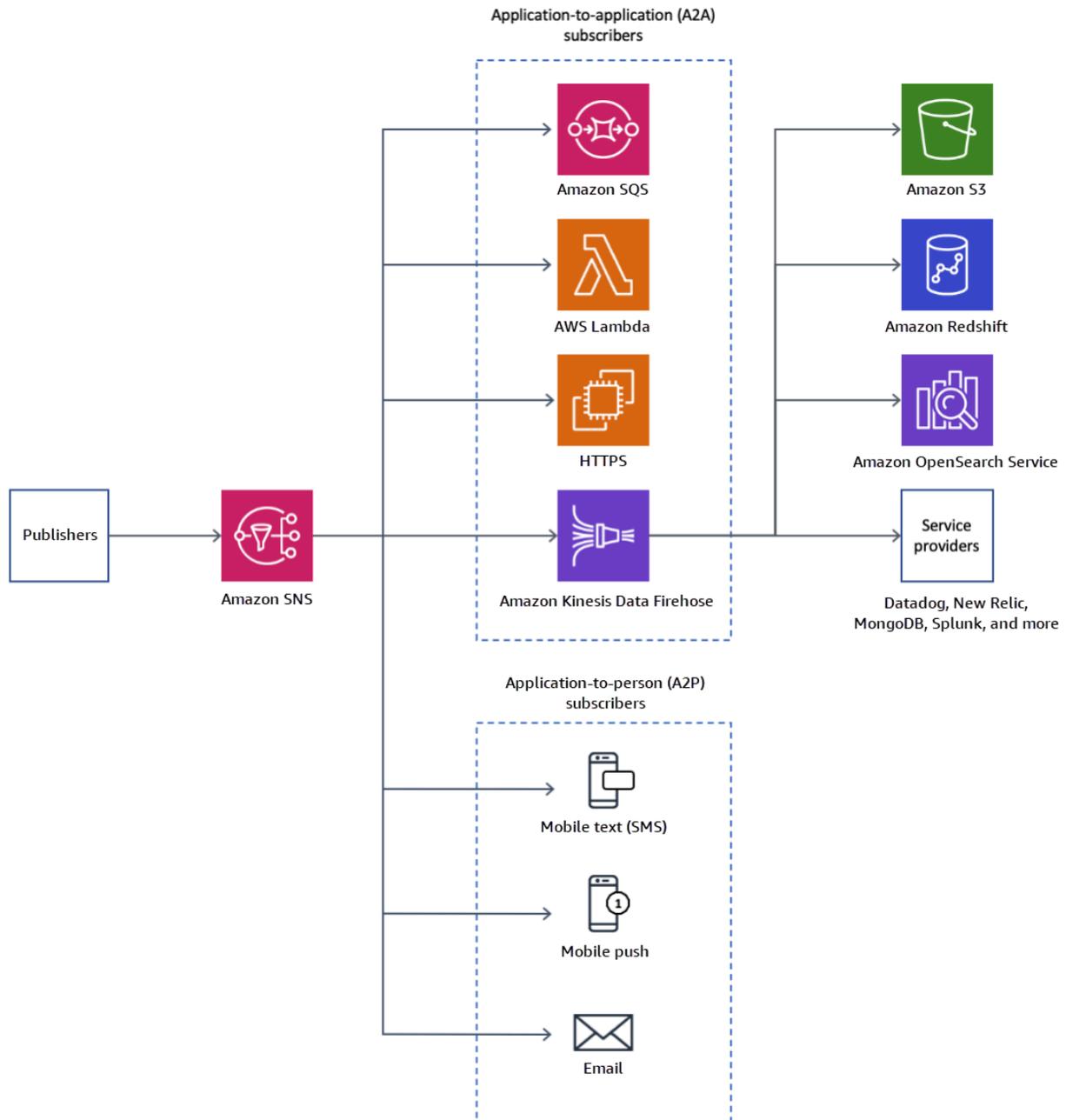
Gerenciamento de identidade e acesso .....	1078
Público .....	1078
Autenticando com identidades .....	1079
Gerenciamento do acesso usando políticas .....	1083
Controle de acesso .....	1085
Visão geral .....	1086
Como o Amazon Simple Notification Service funciona com o IAM .....	1106
Ações de políticas .....	1107
atributos de políticas .....	1108
Chaves de condição de políticas .....	1108
ACLs .....	1109
ABAC .....	1110
Credenciais temporárias .....	1110
Permissões de entidade principal .....	1111
Perfis de serviço .....	1111
Perfis vinculados ao serviço .....	1112
Exemplos de políticas baseadas em identidade .....	1112
Políticas baseadas em identidade .....	1116
Políticas baseadas em recursos .....	1117
Usar políticas baseadas em identidade .....	1118
Usar credenciais temporárias .....	1125
Referência de permissões da API .....	1126
Registro e monitoramento .....	1130
Registrar em log chamadas de API usando o CloudTrail .....	1131
Monitorar tópicos usando o CloudWatch .....	1140
Validação de conformidade .....	1157
Resiliência .....	1158
Segurança da infraestrutura .....	1158
Práticas recomendadas .....	1159
Melhores práticas preventivas .....	1159
Solução de problemas .....	1164
Solução de problemas com tópicos usando o X-Ray .....	1164
Rastreamento ativo .....	1164
Permissões .....	1165
Habilitar o rastreamento ativo .....	1165
Habilitar o rastreamento ativo em um tópico do Amazon SNS (AWS SDK) .....	1166

---

Habilitar o rastreamento ativo em um tópico do Amazon SNS (AWS CLI) .....	1167
Habilitar o rastreamento ativo em um tópico do Amazon SNS (AWS CloudFormation) .....	1167
Verificar se o rastreamento ativo está habilitado .....	1167
Testar .....	1168
Histórico de documentação .....	1170
Glossário do AWS .....	1179
.....	mclxxx

# O que é o Amazon SNS?

O Amazon Simple Notification Service (Amazon SNS) é um serviço gerenciado que fornece entrega de mensagens de editores para assinantes (também conhecido como Produtores e Consumidores). Os editores se comunicam de maneira assíncrona com os assinantes produzindo e enviando mensagens para um tópico, que é um canal de comunicação e um ponto de acesso lógico. Os clientes podem se inscrever no tópico do SNS e receber mensagens publicadas usando um tipo de endpoint compatível, como Amazon Data Firehose, Amazon SQS, HTTP, e-mail AWS Lambda, notificações push móveis e mensagens de texto móveis (SMS).



## Tópicos

- [Recursos e capacidades](#)
- [Serviços relacionados](#)
- [Acessando o Amazon SNS](#)

- [Preços do Amazon SNS](#)
- [Cenários comuns do Amazon SNS](#)
- [Usando o Amazon SNS com um SDK AWS](#)

## Recursos e capacidades

O Amazon SNS fornece os seguintes recursos e funcionalidades:

- Uma pplication-to-application mensagem

Um pplication-to-application sistema de mensagens oferece suporte a assinantes, como fluxos de entrega do Amazon Data Firehose, funções Lambda, filas do Amazon SQS, endpoints HTTP/S e pipelines de eventos fork. AWS Para ter mais informações, consulte [Sistema de mensagens de aplicação para aplicação \(A2A\)](#).

- Uma pplication-to-person notificação

pplication-to-person As notificações fornecem notificações de usuários aos assinantes, como aplicativos móveis, números de telefone celular e endereços de e-mail. Para ter mais informações, consulte [Sistema de mensagens de aplicação para pessoa \(A2P\)](#).

- Tópicos padrão e FIFO

Use um tópico FIFO para garantir uma ordenação estrita de mensagens, para definir grupos de mensagens e para evitar a duplicação de mensagens. Você pode usar filas FIFO e padrão para assinar um tópico FIFO. Para ter mais informações, consulte [Ordenação de mensagens e deduplicação \(tópicos FIFO\)](#).

Use um tópico padrão quando a ordem de entrega de mensagens e a possível duplicação de mensagens não forem críticas. Todos os protocolos de entrega suportados podem se inscrever em um tópico padrão.

- Durabilidade das mensagens

O Amazon SNS usa várias estratégias que funcionam juntas para fornecer durabilidade das mensagens:

- As mensagens publicadas são armazenadas em vários servidores e datacenters geograficamente separados.
- Se um endpoint inscrito não estiver disponível, o Amazon SNS executará um [Política de novas tentativas de entrega](#).

- Para preservar todas as mensagens que não são entregues antes do término da política de repetição de entrega, você pode criar um [Fila de mensagens mortas](#).
- Arquivamento, reprodução e análise de mensagens

Você pode arquivar mensagens com o Amazon SNS de várias maneiras, incluindo a inscrição de [streams de entrega do Firehose em tópicos do SNS, o que permite enviar notificações para endpoints de análise](#), como buckets do Amazon Simple Storage Service (Amazon S3), tabelas do Amazon Redshift e muito mais. Além disso, os tópicos FIFO do Amazon SNS comportam arquivamento e reprodução de mensagens como um arquivamento de mensagens local, sem código, que permite aos proprietários de tópicos armazenar (ou arquivar) mensagens no respectivo tópico. Os assinantes de tópicos podem então recuperar (ou reproduzir) as mensagens arquivadas de volta em um endpoint inscrito. Para saber mais, consulte [Arquivamento e reprodução de mensagens de tópicos FIFO](#).

- Atributos de mensagens

Os atributos de mensagem permitem fornecer quaisquer metadados arbitrários sobre a mensagem. [the section called “Atributos de mensagens”](#).

- Filtragem de mensagens

Por padrão, cada assinante recebe todas as mensagens publicadas no tópico. Para receber um subconjunto de mensagens, um assinante deve atribuir uma política de filtro à assinatura do tópico. Um assinante também pode definir o escopo da política de filtro para habilitar a filtragem baseada em carga útil ou baseada em atributo. O valor padrão para o escopo da política de filtro é `MessageAttributes`. Quando os atributos de mensagem de entrada correspondem aos atributos de política de filtro, a mensagem é entregue ao endpoint inscrito. Caso contrário, a mensagem será filtrada. Quando o escopo da política de filtro é `MessageBody`, os atributos da política de filtro são comparados com a carga útil. Para ter mais informações, consulte [Filtragem de mensagens](#).

- Segurança de mensagens

A criptografia do lado do servidor protege o conteúdo das mensagens que são armazenadas nos tópicos do Amazon SNS, usando chaves de criptografia fornecidas por AWS KMS. Para ter mais informações, consulte [the section called “Criptografia inativa”](#).

Você também pode estabelecer uma conexão privada entre o Amazon SNS e a nuvem privada virtual (VPC). Para obter mais informações, consulte [the section called “Privacidade do tráfego entre redes”](#).

## Serviços relacionados

Você pode usar os seguintes serviços com o Amazon SNS:

- O Amazon SQS oferece uma fila hospedada segura, durável e disponível que permite integrar e desacoplar sistemas de software e componentes distribuídos. O Amazon SQS está relacionado ao Amazon SNS das seguintes maneiras:
  - O Amazon SNS [Filas de mensagens mortas](#) do habilitado pelo Amazon SQS para mensagens não entregues.
  - Você pode [assinar uma fila do Amazon SQS em um tópico do Amazon SNS](#).
  - Você pode inscrever uma [fila FIFO](#) ou uma [fila padrão](#) do Amazon SQS em um [tópico FIFO do Amazon SNS](#). Somente as filas FIFO do Amazon SQS garantem que as mensagens sejam recebidas em ordem e sem duplicatas.
- O AWS Lambda permite a criação de aplicações que respondam rapidamente a novas informações. Execute o código de sua aplicação em funções do Lambda em infraestrutura de computação altamente disponível. Para mais informações, consulte o [Guia do desenvolvedor do AWS Lambda](#). Você pode [inscrever uma função do Lambda em um tópico do SNS](#).
- AWS Identity and Access Management (IAM) ajuda você a controlar com segurança o acesso aos AWS recursos para seus usuários. Use o IAM para controlar quem pode usar os tópicos do Amazon SNS (autenticação) e quais tópicos eles podem usar e de que maneira (autorização). Para ter mais informações, consulte [Usar políticas baseadas em identidade com o Amazon SNS](#).
- AWS CloudFormation permite que você modele e configure seus AWS recursos. Crie um modelo que descreva os AWS recursos que você deseja, incluindo tópicos e assinaturas do Amazon SNS. AWS CloudFormation cuida do provisionamento e da configuração desses recursos para você. Para mais informações, consulte o [Guia do usuário do AWS CloudFormation](#).

## Acessando o Amazon SNS

Você pode configurar e gerenciar tópicos e assinaturas do SNS usando o console do Amazon SNS, as ferramentas de linha de comando ou os SDKs. AWS

- O [Console do Amazon SNS](#) fornece uma interface de usuário conveniente para criar tópicos e assinaturas, enviar e receber mensagens e monitorar eventos e logs.

- O AWS Command Line Interface (AWS CLI) fornece acesso direto à API do Amazon SNS para casos de uso avançados de configuração e automação. Para obter mais informações, consulte [Usar o Amazon SNS com o AWS CLI](#).
- AWS fornece SDKs em vários idiomas. Para obter mais informações, consulte [SDKs e toolkits](#).

## Preços do Amazon SNS

O Amazon SNS não tem custos iniciais. Você paga com base no número de mensagens publicadas, no número de notificações que você entrega e em quaisquer chamadas adicionais à API para gerenciar tópicos e assinaturas. Os preços de entrega variam de acordo com o tipo de endpoint. Você pode começar gratuitamente com o nível gratuito do Amazon SNS.

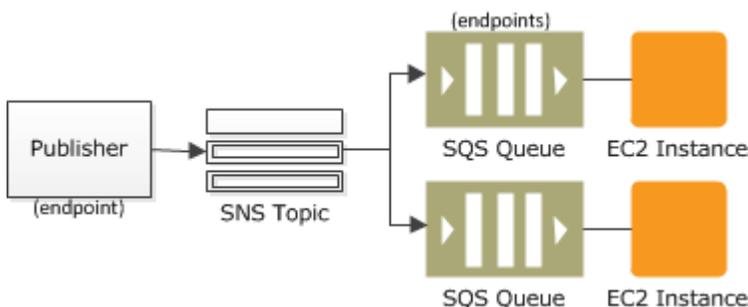
Para obter mais informações, consulte [Definição de preço do Amazon SNS](#).

## Cenários comuns do Amazon SNS

### Integração de aplicações

O cenário do Fanout é quando uma mensagem publicada em um tópico do SNS é replicada e enviada para vários endpoints, como streams de entrega do Firehose, filas do Amazon SQS, endpoints HTTP (S) e funções Lambda. Isso permite o processamento paralelo assíncrono.

Por exemplo, você pode desenvolver uma aplicação que publica uma mensagem em um tópico do SNS sempre que um pedido de um produto é feito. Assim, as filas do SQS inscritas no tópico do SNS receberão notificações idênticas para o novo pedido. Uma instância de servidor do Amazon Elastic Compute Cloud (Amazon EC2) anexada a uma das filas do SQS pode lidar com o processamento ou o preenchimento do pedido. E você pode anexar outra instância de servidor do Amazon EC2 a um data warehouse para análise de todos os pedidos recebidos.



Também é possível usar “fanout” para replicar os dados enviados ao seu ambiente e produção com seu ambiente de teste. Expandindo o exemplo anterior, você pode inscrever mais uma fila do SQS

para o mesmo tópico do SNS para novos pedidos. Ao anexar essa nova fila do SQS ao seu ambiente de teste, você pode continuar a melhorar e testar seu aplicativo usando os dados recebidos do seu ambiente de produção.

### Important

Certifique-se de considerar a privacidade e a segurança dos dados antes de enviar quaisquer dados de produção para o seu ambiente de teste.

Para obter mais informações, consulte os seguintes recursos do :

- [Fanout para streams de entrega do Firehose](#)
- [Fanout para funções do Lambda](#)
- [Fanout para filas do Amazon SQS](#)
- [Fanout para endpoints HTTP\(S\)](#)
- [Computação orientada a eventos com o Amazon SNS AWS e serviços de computação, armazenamento, banco de dados e rede](#)

## Alertas do

Alertas de aplicações e do sistema são notificações que são acionadas por limites predefinidos. O Amazon SNS pode enviar essas notificações para usuários especificados via SMS e e-mail. Por exemplo, você pode receber uma notificação imediata quando ocorrer um evento, como uma alteração específica no seu grupo do Amazon EC2 Auto Scaling, um novo arquivo carregado em um bucket do Amazon S3 ou um limite métrico violado na Amazon. CloudWatch Para obter mais informações, consulte [Configuração de notificações do Amazon SNS no Guia CloudWatch](#) do usuário da Amazon.

## Notificações ao usuário

O Amazon SNS pode enviar mensagens de e-mail de push e mensagens de texto (mensagens SMS) para indivíduos ou grupos. Por exemplo, você pode enviar confirmações de pedidos de comércio eletrônico como notificações do usuário. Para obter mais informações sobre o uso do Amazon SNS para enviar mensagens SMS, consulte [Mensagens de texto em dispositivos móveis \(SMS\)](#).

## Notificações por push para dispositivos móveis

Notificações por push para dispositivos móveis permitem que você envie mensagens diretamente para aplicativos móveis. Por exemplo, você pode usar o Amazon SNS para enviar notificações de atualização para um aplicativo. A mensagem de notificação pode incluir um link para fazer download e instalar a atualização. Para obter mais informações sobre como usar o Amazon SNS para enviar mensagens de notificação por push, consulte [Notificações por push para dispositivos móveis](#).

## Usando o Amazon SNS com um SDK AWS

AWS kits de desenvolvimento de software (SDKs) estão disponíveis para muitas linguagens de programação populares. Cada SDK fornece uma API, exemplos de código e documentação que facilitam a criação de aplicações em seu idioma preferido pelos desenvolvedores.

Documentação do SDK	Exemplos de código
<a href="#">AWS SDK for C++</a>	<a href="#">AWS SDK for C++ exemplos de código</a>
<a href="#">AWS CLI</a>	<a href="#">AWS CLI exemplos de código</a>
<a href="#">AWS SDK for Go</a>	<a href="#">AWS SDK for Go exemplos de código</a>
<a href="#">AWS SDK for Java</a>	<a href="#">AWS SDK for Java exemplos de código</a>
<a href="#">AWS SDK for JavaScript</a>	<a href="#">AWS SDK for JavaScript exemplos de código</a>
<a href="#">AWS SDK para Kotlin</a>	<a href="#">AWS SDK para Kotlin exemplos de código</a>
<a href="#">AWS SDK for .NET</a>	<a href="#">AWS SDK for .NET exemplos de código</a>
<a href="#">AWS SDK for PHP</a>	<a href="#">AWS SDK for PHP exemplos de código</a>
<a href="#">AWS Tools for PowerShell</a>	<a href="#">Ferramentas para exemplos PowerShell de código</a>
<a href="#">AWS SDK for Python (Boto3)</a>	<a href="#">AWS SDK for Python (Boto3) exemplos de código</a>
<a href="#">AWS SDK for Ruby</a>	<a href="#">AWS SDK for Ruby exemplos de código</a>

Documentação do SDK	Exemplos de código
<a href="#">AWS SDK para Rust</a>	<a href="#">AWS SDK para Rust exemplos de código</a>
<a href="#">SDK da AWS para SAP ABAP</a>	<a href="#">SDK da AWS para SAP ABAP exemplos de código</a>
<a href="#">AWS SDK for Swift</a>	<a href="#">AWS SDK for Swift exemplos de código</a>

Para obter exemplos específicos do Amazon SNS, consulte [Exemplos de código para Amazon SNS usando SDKs AWS](#).

 Exemplo de disponibilidade

Você não consegue encontrar o que precisa? Solicite um código de exemplo no link Fornecer feedback na parte inferior desta página.

# Origens e destinos de eventos do Amazon SNS

O Amazon SNS pode receber notificações orientadas por eventos de muitas fontes da AWS e notificações de “fanout” para destinos de aplicação para aplicação (A2A) e aplicação para pessoa (A2P). Esta seção lista origens e destinos de eventos compatíveis e fornece links para obter mais informações.

## Tópicos

- [Fontes de eventos do Amazon SNS](#)
- [Destinos de eventos do Amazon SNS](#)

## Fontes de eventos do Amazon SNS

Esta página lista os serviços da AWS que podem publicar eventos em tópicos do Amazon SNS, agrupados pelas respectivas [categorias de produtos da AWS](#).

### Note

O Amazon SNS introduziu os [tópicos FIFO](#) em outubro de 2020. Atualmente, a maioria dos serviços da AWS oferecem suporte ao envio de eventos apenas para tópicos padrão.

## Serviços de análise

Serviço da AWS	Benefício do uso com o Amazon SNS
<a href="#">Amazon Athena</a> : permite analisar dados no Amazon S3 utilizando o SQL padrão.	Receba notificações se os limites de controle forem excedidos. Para obter mais informações, consulte <a href="#">Setting Data Usage Control Limits</a> (“Definir limites de controle de uso de dados”) no Manual do usuário do Amazon Athena.
<a href="#">AWS Data Pipeline</a> : ajuda a automatizar a movimentação e a transformação de dados.	Receba notificações sobre o status dos componentes do pipeline. Para obter mais informações, consulte <a href="#">SnsAlarm</a> no AWS Data Pipeline Guia do desenvolvedor.

Serviço da AWS	Benefício do uso com o Amazon SNS
<p><a href="#">Amazon Redshift</a>: gerencia todo o trabalho de configuração, operação e escalabilidade de um data warehouse.</p>	<p>Receba notificações de eventos do Amazon Redshift. Para obter mais informações, consulte <a href="#">Notificações de eventos do Amazon Redshift</a> no Guia de gerenciamento do Amazon Redshift.</p>

## Serviços de integração de aplicações

Serviço da AWS	Benefício do uso com o Amazon SNS
<p><a href="#">Amazon EventBridge</a> — entrega um fluxo de dados em tempo real de seus próprios aplicativos, aplicativos software-as-a-service (SaaS) e AWS serviços e encaminha esses dados para destinos, incluindo o Amazon SNS. EventBridge era anteriormente chamado de CloudWatch Eventos.</p>	<p>Receba notificações de EventBridge eventos. Para obter mais informações, consulte as <a href="#">EventBridge metas da Amazon</a> no Guia EventBridge do usuário da Amazon.</p>
<p><a href="#">AWS Step Functions</a>: permite combinar funções do AWS Lambda e outros serviços da AWS para construir aplicações essenciais aos negócios.</p>	<p>Receba notificação de eventos do Step Functions. Para obter mais informações, consulte <a href="#">Call Amazon SNS with Step Functions</a> (“Chamar o Amazon SNS com Step Functions”) no Guia do desenvolvedor do AWS Step Functions.</p>

## Serviços de faturamento e gerenciamento de custos

Serviço da AWS	Benefício do uso com o Amazon SNS
<p><a href="#">AWS Billing and Cost Management</a>: fornece recursos que ajudam a monitorar os custos e pagar sua fatura.</p>	<p>Receba notificações sobre orçamento e alteração de preços, além de alertas sobre anomalias. Para obter mais informações,</p>

Serviço da AWS	Benefício do uso com o Amazon SNS
	<p>consulte as seguintes páginas do Manual do usuário do AWS Billing:</p> <ul style="list-style-type: none"> <li>• <a href="#">Criar um tópico do Amazon SNS para notificações de orçamento</a></li> <li>• <a href="#">Configuração de notificações</a></li> <li>• <a href="#">Detecção de gastos incomuns com o AWS Cost Anomaly Detection</a></li> </ul>

## Serviços de aplicações de negócios

Serviço da AWS	Benefício do uso com o Amazon SNS
<p><a href="#">Amazon Chime</a>: permite conhecer, conversar e fazer chamadas de negócios dentro e fora da sua organização.</p>	<p>Receba notificações sobre eventos de reunião importantes. Para obter mais informações, consulte <a href="#">Amazon Chime SDK event notifications</a> (“Notificações de eventos do SDK do Amazon Chime”) no Guia do desenvolvedor do Amazon Chime.</p>

## Serviços de computação

Serviço da AWS	Benefício do uso com o Amazon SNS
<p><a href="#">Amazon EC2 Auto Scaling</a>: ajuda a ter o número correto de instâncias do Amazon Elastic Compute Cloud (Amazon EC2) disponíveis para processar a carga da aplicação.</p>	<p>Receba notificações quando o Auto Scaling iniciar ou encerrar instâncias do Amazon EC2 em seu grupo do Auto Scaling. Para obter mais informações, consulte <a href="#">Obter notificações do Amazon SNS quando o grupo de Auto Scaling é dimensionado</a> no Guia do usuário do Amazon EC2 Auto Scaling.</p>

Serviço da AWS	Benefício do uso com o Amazon SNS
<p><a href="#">EC2 Image Builder</a> — ajuda a automatizar a criação, o gerenciamento e a implantação de imagens personalizadas, seguras e de servidor pré-instaladas up-to-date e pré-configuradas com software e configurações para atender a padrões de TI específicos.</p>	<p>Receba notificações quando as compilações estiverem concluídas. Para obter mais informações, consulte <a href="#">Tracking the latest server images in Amazon EC2 Image Builder pipelines</a> (“Rastreado as imagens de servidor mais recentes nos pipelines do EC2 Image Builder”) no AWS Compute Blog.</p>
<p><a href="#">AWS Elastic Beanstalk</a>: gerencia os detalhes de provisionamento de capacidade, balanceamento de carga, escalabilidade e monitoramento da integridade da aplicação.</p>	<p>Receba notificações sobre eventos importantes que afetam sua aplicação. Para obter mais informações, consulte <a href="#">Notificações do ambiente do Elastic Beanstalk com o Amazon SNS</a> no Guia do desenvolvedor do AWS Elastic Beanstalk.</p>
<p><a href="#">AWS Lambda</a>: permite executar código sem provisionar ou gerenciar servidores.</p>	<p>Receba dados de saída de funções definindo um tópico do SNS como uma fila de mensagens mortas do Lambda ou um destino do Lambda. Para obter mais informações, consulte <a href="#">Invocação assíncrona</a> no Guia do desenvolvedor do AWS Lambda.</p>
<p><a href="#">Amazon Lightsail</a>: ajuda os desenvolvedores a começar a usar a AWS para criar sites ou aplicações da Web.</p>	<p>Receba notificações quando uma métrica de uma de suas instâncias, bancos de dados ou balanceadores de carga ultrapassar um limite especificado. Para obter mais informações, consulte <a href="#">Adicionar contatos de notificação no Amazon Lightsail</a> no Guia do desenvolvedor do Amazon Lightsail.</p>

## Serviços de contêineres

Serviço da AWS	Benefício do uso com o Amazon SNS
<p><a href="#">Amazon EKS Distro</a>: permite criar clusters confiáveis e seguros onde quer que suas aplicações sejam implantadas.</p>	<p>Rastreie as atualizações e os patches de segurança dos clusters criados com o Amazon EKS Distro. Para obter mais informações, consulte <a href="#">Introducing Amazon EKS Distro - an open source Kubernetes distribution used by Amazon EKS</a> (“Apresentando o Amazon EKS Distro - uma distribuição de código aberto do Kubernetes usada pelo Amazon EKS”).</p>
<p><a href="#">Amazon Elastic Container Service (Amazon ECS)</a>: permite executar, interromper e gerenciar os contêineres em um cluster.</p>	<p>Receba notificações quando uma nova AMI otimizada para o Amazon ECS estiver disponível. Para obter mais informações, consulte <a href="#">Subscribing to Amazon ECS-optimized AMI update notifications</a> (“Assinar notificações sobre atualizações da AMI otimizada para Amazon ECS”) no Guia do desenvolvedor do Amazon Elastic Container Service.</p>

## Serviços de envolvimento com o cliente

Serviço da AWS	Benefício do uso com o Amazon SNS
<p><a href="#">Amazon Connect</a>: permite configurar uma central de atendimento em nuvem omnichannel para interagir com seus clientes.</p>	<p>Receba alertas e validações. Para obter mais informações, consulte <a href="#">O poder da AWS com Amazon Connect</a> no Guia do administrador do Amazon Connect.</p>
<p><a href="#">Amazon Pinpoint</a>: ajuda você a envolver seus clientes enviando e-mails, SMS e mensagens de voz, além de notificações por push.</p>	<p>Configure o SMS bidirecional, que permite receber mensagens dos clientes. Para obter mais informações, consulte <a href="#">Using two-way SMS messaging in Amazon Pinpoint</a> (“Usar o sistema de mensagens SMS bidirecionais no</p>

Serviço da AWS	Benefício do uso com o Amazon SNS
<p><a href="#">Amazon Simple Email Service (Amazon SES)</a>: oferece uma forma econômica de enviar e receber e-mails usando seus próprios endereços de e-mail e domínios.</p>	<p>Receba notificações sobre entregas, devoluções e reclamações. Para obter mais informações sobre o controle de acesso do Amazon SNS, consulte <a href="#">Configuring Amazon SNS notifications for Amazon SES</a> (“Casos de exemplo para o controle de acesso do Amazon SNS”) no Guia do desenvolvedor do Amazon Simple Notification Service.</p>

## Serviços de banco de dados

Serviço da AWS	Benefício do uso com o Amazon SNS
<p><a href="#">AWS Database Migration Service</a>: migra dados de bancos de dados locais para a Nuvem AWS.</p>	<p>Receba notificações quando ocorrerem eventos do AWS DMS, por exemplo, quando uma instância de replicação for criada ou excluída. Para obter mais informações, consulte <a href="#">Trabalhar com eventos e notificações no AWS Database Migration Service</a> no Guia do usuário do AWS Database Migration Service.</p>
<p><a href="#">Amazon DynamoDB</a>: fornece performance rápida e previsível com escalabilidade integrada no serviço de banco de dados NoSQL totalmente gerenciado.</p>	<p>Receba notificações quando ocorrerem eventos de manutenção. Para obter mais informações, consulte <a href="#">Personalização das configurações do cluster DAX</a> no Guia do desenvolvedor do Amazon DynamoDB.</p>
<p><a href="#">Amazon ElastiCache</a> — Fornece um cache na memória de alto desempenho, redimensionável e econômico, ao mesmo tempo em que remove a complexidade associada à implantação e</p>	<p>Receba notificações quando ocorrerem eventos significativos. Para obter mais informações, consulte <a href="#">Notificações de eventos</a></p>

Serviço da AWS	Benefício do uso com o Amazon SNS
ao gerenciamento de um ambiente de cache distribuído.	<a href="#">e Amazon SNS no Guia</a> do usuário do Amazon ElastiCache for Memcached.
<a href="#">Amazon Neptune</a> : permite criar e executar aplicações que funcionam com conjuntos de dados altamente conectados.	Receba notificações quando ocorrer um evento do Neptune. Para obter mais informações, consulte <a href="#">Using Neptune event notification</a> (“Usar a notificação de eventos do Neptune”) no Manual do usuário do Neptune.
<a href="#">Amazon Redshift</a> : gerencia todo o trabalho de configuração, operação e escalabilidade de um data warehouse.	Receba notificações de eventos do Amazon Redshift. Para obter mais informações, consulte <a href="#">Notificações de eventos do Amazon Redshift</a> no Guia de gerenciamento do Amazon Redshift.
<a href="#">Amazon Relational Database Service</a> : facilita a configuração, a operação e a escalabilidade de um banco de dados relacional na Nuvem AWS.	Receba notificações de eventos do Amazon RDS. Para obter mais informações, consulte <a href="#">Usar a notificação de evento do Amazon RDS</a> no Manual do usuário do Amazon RDS.

## Serviços das ferramentas de desenvolvedor

Serviço da AWS	Benefício do uso com o Amazon SNS
<a href="#">AWS CodeBuild</a> : compila o código-fonte, executa testes de unidade e produz artefatos prontos para implantação.	Receba notificações se as compilações forem bem-sucedidas, falharem ou passarem de uma fase de compilação para outra. Para obter mais informações, consulte <a href="#">Exemplo de criação de notificações CodeBuild</a> no Guia AWS CodeBuild do usuário.
<a href="#">AWS CodeCommit</a> : fornece controle de versões para armazenar e gerenciar ativos de maneira privativa na nuvem.	Receba notificações sobre eventos CodeCommit do repositório. Para obter mais informações, consulte <a href="#">Exemplo: criar um acionador do AWS CodeCommit para um</a>

Serviço da AWS	Benefício do uso com o Amazon SNS
	<a href="#">tópico do Amazon SNS</a> no Guia do usuário do AWS CodeCommit.
<a href="#">AWS CodeDeploy</a> : automatiza as implantações de aplicações em instâncias do Amazon EC2, em instâncias on-premises, em funções do Lambda sem servidor ou em serviços do Amazon ECS.	Receba notificações sobre CodeDeploy implantações ou eventos de instância. Para obter mais informações, consulte <a href="#">Criar um gatilho para um CodeDeploy evento</a> no Guia AWS CodeDeploy do usuário.
<a href="#">Amazon CodeGuru</a> — Coleta dados de desempenho em tempo de execução de seus aplicativos ativos e fornece recomendações que podem ajudá-lo a ajustar o desempenho do seu aplicativo.	Receba notificações quando ocorrerem anomalias. Para obter mais informações, consulte <a href="#">Trabalho com anomalias e relatórios de recomendação no Guia CodeGuru</a> do usuário da Amazon.
<a href="#">AWS CodePipeline</a> : automatiza as etapas necessárias para lançar alterações de software de maneira contínua.	Receba notificações sobre ações de aprovação . Para obter mais informações, consulte <a href="#">Gerenciar ações de aprovação CodePipeline no</a> Guia AWS CodePipeline do usuário.
<a href="#">AWS CodeStar</a> : crie, gerencie e trabalhe com projetos de desenvolvimento de software na AWS.	Receba notificações sobre os eventos ocorridos nos recursos utilizados por você. Para obter mais informações, consulte <a href="#">Configure Amazon SNS topics for notifications</a> (“Configurar tópicos do Amazon SNS para notificações”) no Manual do usuário do console do Developer Tools.

## Serviços Web e móveis de front-end

Serviço da AWS	Benefício do uso com o Amazon SNS
<a href="#">Amazon Pinpoint</a> : ajuda você a envolver seus clientes enviando e-mails, SMS e mensagens de voz, além de notificações por push.	Configure o SMS bidirecional, que permite receber mensagens dos clientes. Para obter mais informações, consulte <a href="#">Using two-way</a>

Serviço da AWS	Benefício do uso com o Amazon SNS
	<p><a href="#">SMS messaging in Amazon Pinpoint</a> (“Usar o sistema de mensagens SMS bidirecionais no Amazon Pinpoint”) no Manual do usuário do Amazon Pinpoint.</p>

## Serviços de desenvolvimento de jogos

Serviço da AWS	Benefício do uso com o Amazon SNS
<p><a href="#">Amazon GameLift</a> — Fornece soluções para hospedar servidores de jogos multijogador baseados em sessões na nuvem, incluindo um serviço totalmente gerenciado para implantação, operação e escalabilidade de servidores de jogos.</p>	<p>Receba notificações sobre eventos de matchmaking e fila. Para obter mais informações, consulte as páginas a seguir:</p> <ul style="list-style-type: none"> <li>• Para notificações de matchmaking, consulte <a href="#">Configurar notificação de FlexMatch evento</a> no Amazon GameLift FlexMatch Developer Guide.</li> <li>• Para notificações de fila, consulte <a href="#">Configurar notificação de evento para colocação de sessão de jogo</a> no Amazon GameLift Developer Guide.</li> </ul>

## Serviços da Internet das Coisas

Serviço da AWS	Benefício do uso com o Amazon SNS
<p><a href="#">AWS IoT Core</a>: fornece os serviços de nuvem que conectam os dispositivos IoT a outros serviços da Nuvem AWS e dispositivos.</p>	<p>Receba notificações sobre eventos do AWS IoT Core. Para obter mais informações, consulte <a href="#">Creating an Amazon SNS rule</a> (“Criar uma regra do Amazon SNS”) no Guia do desenvolvedor do AWS IoT.</p>

Serviço da AWS	Benefício do uso com o Amazon SNS
<p><a href="#">AWS IoT Device Defender</a>: permite auditar a configuração de seus dispositivos, monitorar os dispositivos conectados para detectar comportamentos anormais e reduzir os riscos de segurança.</p>	<p>Receba alarmes se um dispositivo violar um comportamento. Para obter mais informações, consulte <a href="#">Como usar a detecção do AWS IoT Device Defender</a> no Guia do desenvolvedor do AWS IoT.</p>
<p><a href="#">AWS IoT Events</a>: permite monitorar os equipamento e as frotas de dispositivos em busca de falhas ou alterações na operação, além de acionar ações caso esses eventos ocorram.</p>	<p>Receba notificações sobre eventos do AWS IoT Events. Para obter mais informações, consulte <a href="#">Amazon Simple Notification Service</a> no Guia do desenvolvedor do AWS IoT Events.</p>
<p><a href="#">AWS IoT Greengrass</a>: estende a AWS a dispositivos físicos para que eles possam atuar localmente nos dados que geram, sem deixar de usar a nuvem para gerenciamento, análise e armazenamento resiliente.</p>	<p>Receba notificações sobre eventos do AWS IoT Greengrass. Para mais informações, consulte <a href="#">SNS connector</a> (“Conector SNS”) no Guia do desenvolvedor do AWS IoT Greengrass Version 1.</p>

## Serviços de machine learning

Serviço da AWS	Benefício do uso com o Amazon SNS
<p><a href="#">Amazon CodeGuru</a> — Coleta dados de desempenho em tempo de execução de seus aplicativos ativos e fornece recomendações que podem ajudá-lo a ajustar o desempenho do seu aplicativo.</p>	<p>Receba notificações quando ocorrerem anomalias. Para obter mais informações, consulte <a href="#">Trabalho com anomalias e relatórios de recomendação no Guia CodeGuru</a> do usuário da Amazon.</p>
<p><a href="#">Amazon DevOps Guru</a> — Gera insights operacionais usando aprendizado de máquina para ajudar você a melhorar o desempenho de seus aplicativos operacionais.</p>	<p>Encaminhe os insights e as confirmações. Para obter mais informações, consulte <a href="#">Forneça insights operacionais baseados em ML para suas equipes de plantão por meio do PagerDuty Amazon DevOps Guru</a> no blog de AWSgerenciamento e governança.</p>

Serviço da AWS	Benefício do uso com o Amazon SNS
<p><a href="#">Amazon Lookout for Metrics</a>: localiza anomalias nos dados, determina as causas raiz e permite tomar medidas rapidamente.</p>	<p>Receba notificações sobre anomalias. Para obter mais informações, consulte <a href="#">Using Amazon SNS with Lookout for Metrics</a> (“Usar o Amazon SNS com Lookout for Metrics”) no Guia do desenvolvedor do Amazon Lookout for Metrics.</p>
<p><a href="#">Amazon Rekognition</a>: permite adicionar análises de imagens e vídeos às aplicações</p>	<p>Receba notificações sobre os resultados de solicitações. Para obter mais informações, consulte <a href="#">Reference: Video analysis results notification</a> (“Referência: notificação sobre resultados da análise de vídeo”) no Guia do desenvolvedor do Amazon Rekognition.</p>
<p><a href="#">Amazon SageMaker</a> — permite que cientistas de dados e desenvolvedores criem e treinem modelos de aprendizado de máquina e, em seguida, os implantem diretamente em um ambiente hospedado pronto para produção.</p>	<p>Receba notificações se um objeto de dados for rotulado. Para obter mais informações, consulte <a href="#">Criação de um trabalho de rotulagem de streaming</a> no Amazon SageMaker Developer Guide.</p>

## Serviços de gerenciamento e governança

Serviço da AWS	Benefício do uso com o Amazon SNS
<p><a href="#">AWS Chatbot</a>— Permite que DevOps as equipes de desenvolvimento de software usem as salas de bate-papo do Amazon Chime e do Slack para monitorar e responder a eventos operacionais na nuvem. AWS</p>	<p>Envie notificações para salas de conversa. Para obter mais informações, consulte <a href="#">Configurar o AWS Chatbot</a> no Guia do administrador do AWS Chatbot.</p>
<p><a href="#">AWS CloudFormation</a>: permite que você crie e provisione implantações de infraestrutura da AWS de maneira previsível e repetível.</p>	<p>Receba notificações quando as pilhas forem criadas e atualizadas. Para obter mais informações, consulte <a href="#">Configurar opções de</a></p>

Serviço da AWS	Benefício do uso com o Amazon SNS
	<p><a href="#">pilha do AWS CloudFormation</a> no Guia do usuário do AWS CloudFormation.</p>
<p><a href="#">AWS CloudTrail</a>: fornece o histórico de eventos da atividade em sua Conta da AWS.</p>	<p>Receba notificações ao CloudTrail publicar novos arquivos de log em seu bucket do Amazon S3. Para obter mais informações, consulte <a href="#">Configuração de notificações CloudTrail do Amazon SNS</a> no Guia AWS CloudTrail do usuário.</p>
<p><a href="#">Amazon CloudWatch</a> — Monitora seus AWS recursos e os aplicativos em que você executa AWS em tempo real.</p>	<p>Receba notificações quando os alarmes mudarem de estado. Para obter mais informações, consulte <a href="#">Usando CloudWatch alarmes da Amazon</a> no Guia do CloudWatch usuário da Amazon.</p>
<p><a href="#">AWS Config</a>: fornece uma visão detalhada da configuração dos recursos da AWS em sua Conta da AWS.</p>	<p>Receba notificações quando os recursos forem atualizados ou quando o AWS Config avaliar as regras gerenciadas ou personalizadas em comparação com seus recursos. Para obter mais informações, consulte <a href="#">Notificações que o AWS Config envia a um tópico do SNS</a> e <a href="#">Exemplo de notificações sobre alteração de itens de configuração</a> no Guia do desenvolvedor do AWS Config.</p>
<p><a href="#">AWS Control Tower</a>: permite configurar e controlar um ambiente multiconta da AWS seguro e compatível.</p>	<p>Use alertas para ajudar a evitar desvios em sua zona de aterrissagem e receba notificações sobre compatibilidade. Para obter mais informações, consulte <a href="#">Tracking alerts through Amazon Simple Notification Service</a> (“Acompanhar alertas pelo Amazon Simple Notification Service”) no Manual do usuário do AWS Control Tower.</p>

Serviço da AWS	Benefício do uso com o Amazon SNS
<p><a href="#">AWS License Manager</a>: ajuda a gerenciar licenças de software de fornecedores de software de maneira centralizada na AWS e em seus ambientes on-premises.</p>	<p>Receba notificações e alertas do License Manager. Para obter mais informações, consulte <a href="#">Configurações no License Manager</a> no Guia do Usuário do License Manager e <a href="#">Criação de ServiceNow incidentes para AWS License Manager notificações</a> no Blog AWS de Gerenciamento e Governança.</p>
<p><a href="#">AWS Service Catalog</a>: permite que administradores de TI criem, gerenciem e distribuam portfólios de produtos aprovados para usuários finais, que podem, então, acessar os produtos de que precisam em um portal personalizado.</p>	<p>Receba notificações sobre eventos de pilha. Para obter mais informações, consulte <a href="#">Limitações das notificações do AWS Service Catalog</a> no Guia do administrador do Service Catalog.</p>
<p><a href="#">AWS Systems Manager</a>: permite visualizar e controlar a infraestrutura na AWS.</p>	<p>Receba notificações sobre o status dos comandos. Para obter mais informações, consulte <a href="#">Monitoring Systems Manager status changes using Amazon SNS notifications</a> (“Monitorar alterações de status do Systems Manager usando as notificações do Amazon SNS”) no Manual do usuário do AWS Systems Manager.</p>

## Serviços de mídia

Serviço da AWS	Benefício do uso com o Amazon SNS
<p><a href="#">Amazon Elastic Transcoder</a>: permite converter arquivos de mídia armazenados no Amazon S3 em arquivos de mídia nos formatos usados em dispositivos de reprodução dos consumidores.</p>	<p>Receba notificações quando os trabalhos mudarem de status. Para obter mais informações, consulte <a href="#">Notifications of job status</a> (“Notificações de status de trabalhos”) no Guia do desenvolvedor do Amazon Elastic Transcoder.</p>

## Serviços de migração e transferência

Serviço da AWS	Benefício do uso com o Amazon SNS
<p><a href="#">AWS Application Discovery Service</a>: ajuda você a planejar a migração para a Nuvem AWS coletando dados de configuração e uso sobre seus servidores on-premises.</p>	<p>Receba notificações sobre eventos por meio do AWS CloudTrail. Para obter mais informações, consulte <a href="#">Registrar chamadas de API do Application Discovery Service com o AWS CloudTrail</a> no Guia do usuário do Application Discovery Service.</p>
<p><a href="#">AWS Database Migration Service</a>: migra dados de bancos de dados locais para a Nuvem AWS.</p>	<p>Receba notificações quando ocorrerem eventos do AWS DMS, por exemplo, quando uma instância de replicação for criada ou excluída. Para obter mais informações, consulte <a href="#">Trabalhar com eventos e notificações no AWS Database Migration Service</a> no Guia do usuário do AWS Database Migration Service.</p>
<p><a href="#">AWS Snowball</a>— Usa dispositivos de armazenamento físico para transferir grandes quantidades de dados entre o Amazon S3 e seu local de armazenamento de dados no local em alta velocidade. faster-than-internet</p>	<p>Receba notificações relacionadas a trabalhos do Snowball. Para obter mais informações, consulte:</p> <ul style="list-style-type: none"> <li>• <a href="#">Snowball notifications</a> (“Notificações do Snowball”) no AWS SnowballManual do usuário do</li> <li>• <a href="#">Step 5: Choose your notification preferences</a> (“Etapa 5: escolha suas preferências de notificação”) no Guia do desenvolvedor do AWS Snowball Edge</li> <li>• <a href="#">Step 5: Choose your notification preferences</a> (“Etapa 5: escolha suas preferências de</li> </ul>

Serviço da AWS	Benefício do uso com o Amazon SNS
	notificação”) no Guia do desenvolvedor do AWS Snowcone

## Serviços de rede e de entrega de conteúdo

Serviço da AWS	Benefício do uso com o Amazon SNS
<p><a href="#">Amazon API Gateway</a> — permite que você crie e implante seus próprios REST e WebSocket APIs em qualquer escala.</p>	<p>Receba mensagens publicadas em um endpoint do API Gateway. Para obter mais informações, consulte <a href="#">Tutorial: Criar uma API REST do API Gateway com integração da AWS</a> no Guia do desenvolvedor do API Gateway.</p>
<p><a href="#">Amazon CloudFront</a> — Acelera a distribuição do seu conteúdo web estático e dinâmico, como arquivos.html, .css, .php, imagens e arquivos de mídia.</p>	<p>Receba notificações quando ocorrerem alarmes com base em CloudFront métricas especificadas. Para obter mais informações, consulte <a href="#">Configuração de alarmes para receber notificações</a> no Amazon CloudFront Developer Guide.</p>
<p><a href="#">AWS Direct Connect</a>: vincula sua rede interna a um local do AWS Direct Connect usando um cabo de fibra óptica Ethernet padrão.</p>	<p>Receba notificações quando houver alteração de estado nos alarmes que monitoram o estado de uma conexão do AWS Direct Connect. Para obter mais informações, consulte <a href="#">Criação de CloudWatch alarmes para monitorar AWS Direct Connect conexões</a> no Guia do AWS Direct Connect usuário.</p>
<p><a href="#">Elastic Load Balancing</a>: distribui automaticamente o tráfego de entrada entre vários destinos, como instâncias do Amazon EC2, contêineres e endereços IP, em uma ou mais zonas de disponibilidade.</p>	<p>Receba notificações sobre os alarmes criados para eventos de balanceador de carga. Para obter mais informações, consulte <a href="#">Criar CloudWatch alarmes para seu balanceador de</a></p>

Serviço da AWS	Benefício do uso com o Amazon SNS
	<a href="#">carga no Guia do usuário para balanceadores de carga clássicos.</a>
<a href="#">Amazon Route 53</a> : fornece registro de domínios, roteamento de DNS e verificação de integridade.	Receba notificações quando o status da verificação e integridade for não íntegro. Para obter mais informações, consulte <a href="#">To receive an Amazon SNS notification when a health check status is unhealthy (console)</a> (“Para receber uma notificação do Amazon SNS quando um status de verificação de integridade for não íntegro (console) no Guia do desenvolvedor do Amazon Route 53.
<a href="#">Amazon Virtual Private Cloud (Amazon VPC)</a> : permite iniciar recursos da AWS em uma rede virtual definida por você.	Receba notificações relacionadas a eventos específicos que ocorrem nos endpoints da interface. Para obter mais informações, consulte <a href="#">Criar e gerenciar uma notificação para um serviço de endpoint</a> no Manual do usuário da Amazon VPC.

## Serviços de segurança, identidade e compatibilidade

Serviço da AWS	Benefício do uso com o Amazon SNS
<a href="#">AWS Directory Service</a> : fornece várias maneiras de usar o Microsoft Active Directory (AD) com outros serviços da AWS.	Receba mensagens de e-mail ou texto (SMS) quando houver alteração no status de seu diretório. Para obter mais informações, consulte <a href="#">Configure directory status notifications</a> (“Configurar notificações de status do diretório”) no Guia de administração do AWS Directory Service.
<a href="#">Amazon GuardDuty</a> — Fornece monitoramento contínuo de segurança para ajudar a identificar atividades inesperadas e potencialmente	Receba notificações sobre os tipos de descoberta recém-lançados, as atualizações nos tipos de descoberta existentes e outras

Serviço da AWS	Benefício do uso com o Amazon SNS
não autorizadas ou maliciosas em seu AWS ambiente.	alterações de funcionalidade. Para obter mais informações, consulte o <a href="#">tópico Assinatura de GuardDuty anúncios do SNS no Guia</a> do usuário da Amazon. GuardDuty
<a href="#">Amazon Inspector</a> : testa a acessibilidade da rede das instâncias do Amazon EC2 e o estado da segurança das aplicações executadas nessas instâncias.	Receba notificações referentes a eventos do Amazon Inspector. Para obter mais informações, consulte <a href="#">Setting up an SNS topic for Amazon Inspector notifications</a> (“Configurar um tópico do SNS para obter notificações do Amazon Inspector”) no Manual do usuário do Amazon Inspector.
<a href="#">AWS Security Hub</a> : automatiza as verificações de segurança da AWS e centraliza os alertas de segurança.	Receba notificações sobre anúncios do AWS Security Hub, incluindo notificações sobre controles ou padrões do AWS Security Hub que foram adicionados, editados ou retirados. Para obter mais informações, consulte <a href="#">Assinar anúncios do AWS Security Hub com o Amazon SNS</a> .

## Serviços sem servidor

Serviço da AWS	Benefício do uso com o Amazon SNS
<a href="#">Amazon DynamoDB</a> : fornece performance rápida e previsível com escalabilidade integrada no serviço de banco de dados NoSQL totalmente gerenciado.	Receba notificações quando ocorrerem eventos de manutenção. Para obter mais informações, consulte <a href="#">Personalização das configurações do cluster DAX</a> no Guia do desenvolvedor do Amazon DynamoDB.
<a href="#">Amazon EventBridge</a> — entrega um fluxo de dados em tempo real de seus próprios aplicativos, aplicativos software-as-a-service (SaaS) e AWS serviços e encaminha esses dados para	Receba notificações de EventBridge eventos. Para obter mais informações, consulte as <a href="#">EventBridgemetas da Amazon</a> no Guia EventBridge do usuário da Amazon.

Serviço da AWS	Benefício do uso com o Amazon SNS
destinos, incluindo o Amazon SNS. EventBridge era anteriormente chamado de CloudWatch Eventos.	
<p><a href="#">AWS Lambda</a>: permite executar código sem provisionar ou gerenciar servidores.</p>	<p>Receba dados de saída de funções definindo um tópico do SNS como uma fila de mensagens mortas do Lambda ou um destino do Lambda. Para obter mais informações, consulte <a href="#">Invocação assíncrona</a> no Guia do desenvolvedor do AWS Lambda.</p>

## Serviços de armazenamento

Serviço da AWS	Benefício do uso com o Amazon SNS
<p><a href="#">AWS Backup</a>: ajuda a centralizar e automatizar o backup de dados entre serviços da AWS na nuvem e on-premises.</p>	<p>Receba notificações sobre eventos do AWS Backup. Para obter mais informações, consulte <a href="#">Usar o Amazon SNS para monitorar eventos do AWS Backup</a> no Guia do desenvolvedor do AWS Backup.</p>
<p><a href="#">Amazon Elastic File System</a>: fornece armazenamento de arquivos para instâncias do Amazon EC2.</p>	<p>Receba notificações sobre os alarmes criados para eventos do Amazon EFS. Para obter mais informações, consulte <a href="#">Automated monitoring tools</a> (“Ferramentas de monitoramento automatizadas”) no Manual do usuário do Amazon Elastic File System.</p>
<p><a href="#">Amazon S3 Glacier</a>: fornece armazenamento para dados usados com pouca frequência.</p>	<p>Defina uma configuração de notificação em um cofre, de maneira que, quando um trabalho for concluído, uma mensagem será enviada para um tópico do SNS. Para obter mais informações, consulte <a href="#">Configuring vault notifications in Amazon S3 Glacier</a> (“Configurar notificações</p>

Serviço da AWS	Benefício do uso com o Amazon SNS
<p><a href="#">Amazon Simple Storage Service (Amazon S3)</a>: fornece armazenamento de objetos.</p>	<p>de cofre no Amazon S3 Glacier”) no Guia do desenvolvedor do Amazon S3 Glacier.</p> <p>Receba notificações quando ocorrerem alterações em um bucket do Amazon S3 ou na rara ocasião em que os objetos não forem replicados para a região de destino. Para obter mais informações, consulte <a href="#">Demonstração: configurar um bucket para notificações (tópico do SNS ou fila do SQS)</a> e <a href="#">Monitoramento do progresso com métricas de replicação e notificações de eventos do Amazon S3</a> no Manual do usuário do Amazon Simple Storage Service.</p>
<p><a href="#">AWS Snowball</a>— Usa dispositivos de armazenamento físico para transferir grandes quantidades de dados entre o Amazon S3 e seu local de armazenamento de dados no local em alta velocidade. faster-than-internet</p>	<p>Receba notificações relacionadas a trabalhos do Snowball. Para obter mais informações, consulte:</p> <ul style="list-style-type: none"><li>• <a href="#">Snowball notifications</a> (“Notificações do Snowball”) no AWS Snowball Manual do usuário do</li><li>• <a href="#">Step 5: Choose your notification preferences</a> (“Etapa 5: escolha suas preferências de notificação”) no Guia do desenvolvedor do AWS Snowball Edge</li><li>• <a href="#">Step 5: Choose your notification preferences</a> (“Etapa 5: escolha suas preferências de notificação”) no Guia do desenvolvedor do AWS Snowcone</li></ul>

## Fontes de eventos adicionais

Origem	Benefício do uso com o Amazon SNS
<a href="#">Atualizações diárias de recursos da AWS</a>	Receba informações detalhadas em tempo hábil sobre lançamentos e atualizações da AWS por meio de um tópico do Amazon SNS. Esses lançamentos incluem endpoints do Amazon VPC Regiões da AWS, Serviços da AWS integrados às Cotas de AWS Serviços, tipos de instância do Amazon EC2, tipos de instância da Amazon SageMaker Nimble Studio, versões do mecanismo de banco de dados do Amazon RDS e versões do Amazon MSK Apache Kafka. Para obter mais informações, consulte <a href="#">Assinar atualizações diárias de recursos da AWS via Amazon SNS</a> no Blog de notícias da AWS.
<a href="#">Intervalos de endereços IP da AWS</a>	Receba notificações sobre alterações nos intervalos de IP da AWS por um tópico do Amazon SNS. Para obter mais informações, consulte <a href="#">Notificações de intervalos de endereços IP da AWS</a> na Referência geral da Amazon Web Services e <a href="#">Assinar alterações de endereços IP públicos da AWS via Amazon SNS</a> no Blog de notícias da AWS.

Consulte as seguintes fontes para obter mais informações sobre a computação baseada em eventos:

- [O que é uma arquitetura orientada a eventos?](#)
- [Computação orientada por eventos com o Amazon SNS e serviços de computação, armazenamento, banco de dados e redes da AWS](#) no Blog de computação da AWS
- [“Enriching Event-Driven Architectures with AWS Event Fork Pipelines”](#) (“Enriquecer arquiteturas orientadas a eventos com o Event Fork Pipelines”) no Blog de computação da AWS

# Destinos de eventos do Amazon SNS

Esta página lista todos os destinos que podem receber informações sobre eventos, agrupados por [mensagens application-to-application \(A2A\)](#) e [notificações application-to-person \(A2P\)](#).

## Note

O Amazon SNS introduziu os [tópicos FIFO](#) em outubro de 2020. Atualmente, a maioria dos serviços da AWS oferecem suporte ao envio de eventos apenas de tópicos padrão do SNS. O Amazon SQS oferece suporte ao recebimento de eventos de tópicos padrão e FIFO do SNS.

## Destinos A2A

Destino do evento	Benefício do uso com o Amazon SNS
<a href="#">Amazon Data Firehose</a>	Entregue eventos para fluxos de entrega para fins de arquivamento e análise. Por meio de fluxos de entrega, você pode entregar eventos para AWS destinos como Amazon Simple Storage Service (Amazon S3), Amazon Redshift e OpenSearch Amazon Service OpenSearch (Service), ou para destinos de terceiros, como Datadog, New Relic, MongoDB e Splunk. Para ter mais informações, consulte <a href="#">Fanout para streams de entrega do Firehose</a> .
<a href="#">AWS Lambda</a>	Entregue eventos a funções para acionar a execução da lógica de negócios personalizada. Para ter mais informações, consulte <a href="#">Fanout para funções do Lambda</a> .
<a href="#">Amazon SQS</a>	Entregue eventos para filas para fins de integração de aplicações. Para ter mais informações, consulte <a href="#">Fanout para filas do Amazon SQS</a> .

Destino do evento	Benefício do uso com o Amazon SNS
AWS Event Fork Pipelines	Entregue eventos para backup e armazenamento de eventos, pesquisa e análise de eventos ou pipelines de repetição de eventos. Para ter mais informações, consulte <a href="#">Fanout para AWS Event Fork Pipelines</a> .
HTTP/S	Entregue eventos para webhooks externos. Para ter mais informações, consulte <a href="#">Fanout para endpoints HTTP(S)</a> .

## Destinos A2P

Destino do evento	Benefício do uso com o Amazon SNS
SMS	Entregue eventos para telefones celulares como mensagens de texto. Para ter mais informações, consulte <a href="#">Mensagens de texto em dispositivos móveis (SMS)</a> .
E-mail	Entregue eventos para caixas de entrada como mensagens de e-mail. Para ter mais informações, consulte <a href="#">Notificações por e-mail</a> .
Endpoint da plataforma	Entregue eventos para telefones celulares como notificações por push nativas. Para ter mais informações, consulte <a href="#">Notificações por push para dispositivos móveis</a> .
<a href="#">AWS Chatbot</a>	Entregue eventos para salas de bate-papo do Amazon Chime ou canais do Slack. Para obter mais informações, consulte as páginas a seguir no Guia do administrador do AWS Chatbot: <ul style="list-style-type: none"> <li></li> </ul>

Destino do evento	Benefício do uso com o Amazon SNS
	<ul style="list-style-type: none"> <li><a href="#">Configuração AWS Chatbot com o Amazon Chime</a></li> <li>• <a href="#">Configurando AWS Chatbot com o Slack</a></li> <li>• <a href="#">Usar o AWS Chatbot com outros serviços da AWS</a></li> </ul>
PagerDuty	<p>Forneça insights operacionais para equipes de plantão. Para obter mais informações, consulte <a href="#">Forneça insights operacionais baseados em ML para suas equipes de plantão por meio do PagerDuty Amazon DevOps Guru</a> no blog de AWSgerenciamento e governança.</p>

### Note

Você pode entregar tanto eventos da AWS nativos quanto eventos personalizados para aplicações de conversa:

- **Eventos da AWS nativos:** você pode usar o AWS Chatbot para enviar eventos da AWS nativos, por meio de tópicos do Amazon SNS, para o Amazon Chime e o Slack. O conjunto suportado de AWS eventos nativos inclui eventos da AWS Billing and Cost Management, AWS Health, AWS CloudFormation CloudWatch, Amazon e muito mais. Para obter mais informações, consulte [Usando AWS Chatbot com outros serviços](#) no Guia do AWS Chatbot Administrador.
- **Eventos personalizados:** você também pode enviar seus eventos personalizados, por meio de tópicos do Amazon SNS, para o Amazon Chime, Slack e Microsoft Teams. Para fazer isso, você publica eventos personalizados em um tópico do SNS, que entrega os eventos para uma função do Lambda inscrita. A função do Lambda usa o webhook da aplicação de conversa para entregar os eventos aos destinatários. Para obter mais informações, consulte [How do I use webhooks to publish Amazon SNS messages to Amazon Chime,](#)

[Slack, or Microsoft Teams?](#) (“Como uso webhooks para publicar mensagens do Amazon SNS no Amazon Chime, Slack ou Microsoft Teams?”).

# Configurar o acesso para o Amazon SNS

Antes de usar o Amazon SNS pela primeira vez, conclua as etapas abaixo.

Tópicos

- [Etapa 1: criar um Conta da AWS e um usuário do IAM](#)
- [Próximas etapas](#)

## Etapa 1: criar um Conta da AWS e um usuário do IAM

Para acessar qualquer AWS serviço, você deve primeiro criar um [Conta da AWS](#). Você pode usar o seu Conta da AWS para visualizar seus relatórios de atividade e uso e para gerenciar a autenticação e o acesso.

### Inscreva-se para um Conta da AWS

Se você não tiver um Conta da AWS, conclua as etapas a seguir para criar um.

Para se inscrever em um Conta da AWS

1. Abra <https://portal.aws.amazon.com/billing/signup>.
2. Siga as instruções on-line.

Parte do procedimento de inscrição envolve receber uma chamada telefônica e digitar um código de verificação no teclado do telefone.

Quando você se inscreve em um Conta da AWS, um usuário Conta da AWS root é criado. O usuário-raiz tem acesso a todos os Serviços da AWS e recursos na conta. Como uma prática recomendada de segurança, atribua o acesso administrativo para um usuário e use somente o usuário-raiz para executar [tarefas que requerem o acesso de usuário-raiz](#).

AWS envia um e-mail de confirmação após a conclusão do processo de inscrição. A qualquer momento, é possível visualizar as atividades da conta atual e gerenciar sua conta acessando <https://aws.amazon.com/> e selecionando Minha conta.

## Criar um usuário com acesso administrativo

Depois de se inscrever em um Conta da AWS, proteja seu usuário Conta da AWS raiz AWS IAM Identity Center, habilite e crie um usuário administrativo para que você não use o usuário root nas tarefas diárias.

### Proteja seu usuário Conta da AWS root

1. Faça login [AWS Management Console](#) como proprietário da conta escolhendo Usuário raiz e inserindo seu endereço de Conta da AWS e-mail. Na próxima página, digite sua senha.

Para obter ajuda ao fazer login usando o usuário-raiz, consulte [Signing in as the root user](#) (Fazer login como usuário-raiz) no Guia do usuário do Início de Sessão da AWS .

2. Habilite a autenticação multifator (MFA) para o usuário-raiz.

Para obter instruções, consulte [Habilitar um dispositivo de MFA virtual para seu usuário Conta da AWS raiz \(console\) no Guia](#) do usuário do IAM.

### Criar um usuário com acesso administrativo

1. Habilitar o IAM Identity Center.

Para obter instruções, consulte [Habilitar AWS IAM Identity Center](#) no Guia do usuário do AWS IAM Identity Center .

2. No Centro de Identidade do IAM, conceda o acesso administrativo para um usuário.

Para ver um tutorial sobre como usar o Diretório do Centro de Identidade do IAM como fonte de identidade, consulte [Configurar o acesso do usuário com o padrão Diretório do Centro de Identidade do IAM](#) no Guia AWS IAM Identity Center do usuário.

### Iniciar sessão como o usuário com acesso administrativo

- Para fazer login com seu usuário do Centro de Identidade do IAM, use a URL de login que foi enviada ao seu endereço de e-mail quando você criou o usuário do Centro do Usuário do IAM.

Para obter ajuda para fazer login usando um usuário do IAM Identity Center, consulte [Como fazer login no portal de AWS acesso](#) no Guia Início de Sessão da AWS do usuário.

## Atribuir acesso para usuários adicionais

1. No Centro de Identidade do IAM, crie um conjunto de permissões que siga as práticas recomendadas de aplicação de permissões com privilégio mínimo.

Para obter instruções, consulte [Create a permission set](#) no Guia do usuário do AWS IAM Identity Center .

2. Atribua usuários a um grupo e, em seguida, atribua o acesso de autenticação única ao grupo.

Para obter instruções, consulte [Add groups](#) no Guia do usuário do AWS IAM Identity Center .

## Próximas etapas

Agora que você está preparado para trabalhar com o Amazon SNS, [comece](#) criando um tópico, criando uma assinatura para o tópico, publicando uma mensagem no tópico e excluindo a assinatura e o tópico.

# Conceitos básicos do Amazon SNS

Esta seção ajudará você a se familiarizar com o Amazon SNS mostrando como gerenciar tópicos, assinaturas e mensagens usando o console do Amazon SNS.

## Tópicos

- [Pré-requisitos](#)
- [Etapa 1: Criar um tópico](#)
- [Etapa 2: criar uma assinatura para o tópico](#)
- [Etapa 3: Publicar uma mensagem no tópico](#)
- [Etapa 4: Excluir a assinatura e o tópico](#)
- [Próximas etapas](#)

## Pré-requisitos

Antes de começar, conclua as etapas em [Configurar o acesso para o Amazon SNS](#).

## Etapa 1: Criar um tópico

1. Faça login no [console do Amazon SNS](#).
2. No painel de navegação à esquerda, selecione Tópicos.
3. Na página Topics (Tópicos), escolha Create topic (Criar tópico).
4. Por padrão, o console cria um tópico FIFO. Escolha Standard (Padrão).
5. Na seção Detalhes, insira um nome para o tópico, como *MyTopic*.
6. Role até o final do formulário e escolha Create topic (Criar tópico).

O console abrirá a página Details (Detalhes) do tópico.

## Etapa 2: criar uma assinatura para o tópico

1. No painel de navegação à esquerda, escolha Assinaturas.
2. Na página Subscriptions (Assinaturas), escolha Create subscription (Criar assinatura).

3. Na página **Create subscription** (Criar assinatura), escolha o campo **Topic ARN** (ARN do tópico) para ver uma lista dos tópicos em sua Conta da AWS.
4. Escolha o tópico que você criou na etapa anterior.
5. Em **Protocolo**, escolha **Email**.
6. Em **Endpoint**, insira um endereço de e-mail que possa receber notificações.
7. Selecione **Criar assinatura**.

O console abrirá a página **Details** (Detalhes) da nova assinatura.

8. Verifique sua caixa de entrada de e-mail e escolha **Confirm subscription** (Confirmar a assinatura) no e-mail de AWS Notifications. O ID do remetente geralmente é "no-reply@sns.amazonaws.com".
9. O Amazon SNS abre seu navegador da Web e exibe uma confirmação de assinatura com seu ID de assinatura.

## Etapa 3: Publicar uma mensagem no tópico

1. No painel de navegação à esquerda, selecione **Tópicos**.
2. Na página **Topics** (Tópicos), selecione o tópico criado anteriormente e selecione **Publish message** (Publicar mensagem).

O console abre a página **Publish message to topic** (Publicar mensagem no tópico).

3. (Opcional) Na seção **Message details** (Detalhes da mensagem), digite o **Subject** (Assunto), como:

```
Hello from Amazon SNS!
```

4. Na seção **Message body** (Corpo da mensagem), escolha **Identical payload for all delivery protocols** (Carga útil idêntica para todos os protocolos de entrega) e, em seguida, introduza um corpo da mensagem, tal como:

```
Publishing a message to an SNS topic.
```

5. Selecione **Publish message** (Publicar mensagem).

A mensagem é publicada no tópico e o console abre a página **Details** (Detalhes).

6. Verifique sua caixa de entrada de e-mail e verifique se você recebeu um e-mail do Amazon SNS com a mensagem publicada.

## Etapa 4: Excluir a assinatura e o tópico

1. No painel de navegação, escolha Subscriptions (Assinaturas).
2. Na página Subscriptions (Assinaturas), escolha uma assinatura confirmada e escolha Delete (Excluir).

### Note

Não é possível excluir uma confirmação pendente. Depois de 48 horas, o Amazon SNS o exclui automaticamente.

3. Na caixa de diálogo Delete subscription (Excluir assinatura), escolha Delete (Excluir).

A assinatura é excluída.

4. No painel de navegação, escolha Topics (Tópicos).
5. Na página Topics (Tópicos), escolha um tópico e, em seguida, escolha Delete (Excluir).

### Important

Ao excluir um tópico, você também exclui todas as inscrições pertencentes ao tópico.

6. Na caixa de *MyTopic* diálogo Excluir tópico, insira delete me e escolha Excluir.

O tópico é excluído.

## Próximas etapas

Agora que você criou um tópico com uma assinatura e enviou mensagens para o tópico, talvez queira experimentar o seguinte:

- Explorar o [Centro do desenvolvedor da AWS](#).
- Saiba como proteger seus dados na seção [Segurança](#).
- Habilite a [criptografia do lado do servidor](#) para um tópico.

- Habilitar a criptografia no lado do servidor para um tópico com uma [fila criptografada do Amazon Simple Queue Service \(Amazon SQS\)](#) inscrita.
- Assine [AWS Event Fork Pipelines](#) em um tópico.

# Configurar o Amazon SNS

Use o [console do Amazon SNS](#) para criar e configurar tópicos e assinaturas do Amazon SNS. Para obter mais informações sobre o Amazon SNS, consulte [O que é o Amazon SNS?](#).

## Tópicos

- [Criar um tópico do Amazon SNS](#)
- [Assinar tópico do Amazon SNS](#)
- [Excluindo um tópico e uma assinatura do Amazon SNS](#)
- [Marcação de tópicos do Amazon SNS](#)

## Criar um tópico do Amazon SNS

Um tópico do Amazon SNS é um ponto de acesso lógico que atua como um canal de comunicação. Um tópico permite agrupar vários endpoints (como Amazon SQS AWS Lambda, HTTP/S ou um endereço de e-mail).

Para transmitir as mensagens de um sistema produtor de mensagem (por exemplo, um site de comércio eletrônico) que trabalha com vários outros serviços que exigem suas mensagens (por exemplo, sistemas de cumprimento e checkout), crie um tópico para o sistema produtor.

A tarefa inicial e mais comum do Amazon SNS é a criação de um tópico. Esta página mostra como você pode usar o AWS Management Console AWS SDK for Java, o e o AWS SDK for .NET para criar um tópico.

Durante a criação, você escolhe um tipo de tópico (padrão ou FIFO) e nomeia o tópico. Depois de criado um tópico, você não pode alterar o tipo ou o nome do tópico. Todas as outras opções de configuração são opcionais durante a criação do tópico e você pode editá-las posteriormente.

### Important

Não inclua informações de identificação pessoal (PII) nem outras informações confidenciais ou sigilosas em nomes de tópicos. Os nomes dos tópicos podem ser acessados por outros Amazon Web Services, incluindo CloudWatch Logs. Os nomes de tópicos não devem ser usados para dados privados ou sigilosos.

## Tópicos

- [Para criar um tópico usando o AWS Management Console](#)
- [Para criar um tópico usando um AWS SDK](#)

## Para criar um tópico usando o AWS Management Console

1. Faça login no console [do Amazon SNS](#).
2. Execute um destes procedimentos:
  - Se nenhum tópico já tiver sido criado Conta da AWS antes, leia a descrição do Amazon SNS na página inicial.
  - Se os tópicos já tiverem sido criados abaixo Conta da AWS do seu, no painel de navegação, escolha Tópicos.
3. Na página Topics (Tópicos), escolha Create topic (Criar tópico).
4. Na página Create topic (Criar tópico), na seção Details (detalhes), faça o seguinte:
  - a. Em Type (Tipo), escolha um tipo de tópico (Standard (Padrão) ou FIFO).
  - b. Insira um Name (Nome) para o tópico. Para um [tópico FIFO](#), adicione .fifo ao final do nome.
  - c. (Opcional) Insira um Display name (Nome para exibição) para o tópico.

### Important

Ao assinar um endpoint de e-mail, a contagem combinada de caracteres para o nome de exibição do tópico do Amazon SNS e o endereço de e-mail do remetente (por exemplo, no-reply@sns.amazonaws.com) não deve exceder 320 caracteres UTF-8. Você pode usar uma ferramenta de codificação de terceiros para verificar o tamanho do endereço de remetente antes de configurar um nome de exibição para seu tópico do Amazon SNS.

- d. (Opcional) Para um tópico FIFO, você pode escolher content-based message deduplication (Desduplicação de mensagens baseada em conteúdo) para habilitar a desduplicação de mensagens padrão. Para ter mais informações, consulte [Desduplicação de mensagens para tópicos FIFO](#).

5. (Opcional) Expanda a seção Encryption (Criptografia) e faça o seguinte. Para ter mais informações, consulte [Criptografia inativa](#).
  - a. Selecione Ativar criptografia.
  - b. Especifique a AWS KMS chave. Para ter mais informações, consulte [Principais termos](#).

Para cada tipo de KMS, são exibidos Description (Descrição), Account (Conta) e KMS ARN (ARN do KMS).

 Important

Se você não for o proprietário do KMS ou se fizer login com uma conta que não tenha as permissões `kms:ListAliases` e `kms:DescribeKey`, não será possível visualizar as informações sobre o KMS no console do Amazon SNS.

Peça ao proprietário do KMS para conceder essas permissões a você. Para obter mais informações, consulte [AWS KMS API Permissions: Actions and Resources Reference](#) (“Permissões da API do KMS: referência de ações e recursos”) no Guia do desenvolvedor do AWS Key Management Service .

- O alias/aws/sns do KMS AWS gerenciado para Amazon SNS (padrão) é selecionado por padrão.

 Note

Lembre-se do seguinte:

- A primeira vez que você usa o AWS Management Console para especificar o KMS AWS gerenciado para o Amazon SNS para um tópico AWS KMS , cria AWS o KMS gerenciado para o Amazon SNS.
  - Como alternativa, na primeira vez que você usa a Publish ação em um tópico com o SSE ativado, AWS KMS cria o KMS AWS gerenciado para o Amazon SNS.
- Para usar um KMS personalizado da sua AWS conta, escolha o campo Chave KMS e, em seguida, escolha o KMS personalizado na lista.

**Note**

Para obter instruções sobre como criar KMSs personalizados, consulte [Criar chaves](#) no Guia do desenvolvedor do AWS Key Management Service .

- Para usar um ARN KMS personalizado da AWS sua conta ou de AWS outra conta, insira-o no campo Chave do KMS.
6. (Opcional) Por padrão, somente o proprietário do tópico pode publicar ou assinar o tópico. Para configurar permissões de acesso adicionais, expanda a seção Política de acesso. Para ter mais informações, consulte [Gerenciamento de identidade e acesso no Amazon SNS](#) e [Casos de exemplo para controle de acesso do Amazon SNS](#).

**Note**

Quando você cria um tópico usando o console, a política padrão usa a chave de condição `aws:SourceOwner`. Essa chave é semelhante a `aws:SourceAccount`.

7. (Opcional) Para configurar como o Amazon SNS repete tentativas de entrega de mensagem com falha, expanda a seção Delivery retry policy (HTTP/S) (Política de repetição de entrega (HTTP/S)). Para ter mais informações, consulte [Novas tentativas de entrega de mensagens do Amazon SNS](#).
8. (Opcional) Para configurar como o Amazon SNS registra a entrega de mensagens CloudWatch, expanda a seção Registro de status de entrega. Para ter mais informações, consulte [Status de entrega de mensagens do Amazon SNS](#).
9. (Opcional) Para adicionar tags de metadados ao tópico, expanda a seção Tags, insira uma Key (chave) e um Value (Valor) (opcional) e escolha Add tag (Adicionar tag). Para ter mais informações, consulte [Marcação de tópicos do Amazon SNS](#).
10. Escolha Criar tópico.

O tópico é criado e a **MyTopic** página é exibida.

O nome do tópico, o ARN, o nome de exibição (opcional) e o ID da AWS conta do proprietário do tópico são exibidos na seção Detalhes.

11. Copie o ARN do tópico para a área de transferência, por exemplo:

```
arn:aws:sns:us-east-2:123456789012:MyTopic
```

## Para criar um tópico usando um AWS SDK

Para usar um AWS SDK, você deve configurá-lo com suas credenciais. Para obter mais informações, consulte [The shared config and credentials files](#) (“Os arquivos compartilhados de configuração e de credenciais”) no Guia de referência de AWS SDKs e ferramentas.

Os exemplos de código a seguir mostram como usar `CreateTopic`.

.NET

AWS SDK for .NET

### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

Crie um tópico com um nome específico.

```
using System;
using System.Threading.Tasks;
using Amazon.SimpleNotificationService;
using Amazon.SimpleNotificationService.Model;

/// <summary>
/// This example shows how to use Amazon Simple Notification Service
/// (Amazon SNS) to add a new Amazon SNS topic.
/// </summary>
public class CreateSNSTopic
{
    public static async Task Main()
    {
        string topicName = "ExampleSNSTopic";

        IAmazonSimpleNotificationService client = new
AmazonSimpleNotificationServiceClient();

        var topicArn = await CreateSNSTopicAsync(client, topicName);
        Console.WriteLine($"New topic ARN: {topicArn}");
    }
}
```

```
    /// <summary>
    /// Creates a new SNS topic using the supplied topic name.
    /// </summary>
    /// <param name="client">The initialized SNS client object used to
    /// create the new topic.</param>
    /// <param name="topicName">A string representing the topic name.</param>
    /// <returns>The Amazon Resource Name (ARN) of the created topic.</
returns>
    public static async Task<string>
CreateSNSTopicAsync(IAmazonSimpleNotificationService client, string topicName)
    {
        var request = new CreateTopicRequest
        {
            Name = topicName,
        };

        var response = await client.CreateTopicAsync(request);

        return response.TopicArn;
    }
}
```

Crie um tópico com um nome e atributos específicos de FIFO e desduplicação.

```
    /// <summary>
    /// Create a new topic with a name and specific FIFO and de-duplication
attributes.
    /// </summary>
    /// <param name="topicName">The name for the topic.</param>
    /// <param name="useFifoTopic">True to use a FIFO topic.</param>
    /// <param name="useContentBasedDeduplication">True to use content-based de-
duplication.</param>
    /// <returns>The ARN of the new topic.</returns>
    public async Task<string> CreateTopicWithName(string topicName, bool
useFifoTopic, bool useContentBasedDeduplication)
    {
        var createTopicRequest = new CreateTopicRequest()
        {
            Name = topicName,
        };
    }
```

```
    if (useFifoTopic)
    {
        // Update the name if it is not correct for a FIFO topic.
        if (!topicName.EndsWith(".fifo"))
        {
            createTopicRequest.Name = topicName + ".fifo";
        }

        // Add the attributes from the method parameters.
        createTopicRequest.Attributes = new Dictionary<string, string>
        {
            { "FifoTopic", "true" }
        };
        if (useContentBasedDeduplication)
        {
            createTopicRequest.Attributes.Add("ContentBasedDeduplication",
"true");
        }
    }

    var createResponse = await
    _amazonSNSClient.CreateTopicAsync(createTopicRequest);
    return createResponse.TopicArn;
}
```

- Para obter detalhes da API, consulte [CreateTopic](#) Referência AWS SDK for .NET da API.

## C++

### SDK for C++

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```
//! Create an Amazon Simple Notification Service (Amazon SNS) topic.
/*!
\param topicName: An Amazon SNS topic name.
```

```

    \param topicARNResult: String to return the Amazon Resource Name (ARN) for the
    topic.
    \param clientConfiguration: AWS client configuration.
    \return bool: Function succeeded.
    */
bool AwsDoc::SNS::createTopic(const Aws::String &topicName,
                              Aws::String &topicARNResult,
                              const Aws::Client::ClientConfiguration
&clientConfiguration) {
    Aws::SNS::SNSClient snsClient(clientConfiguration);

    Aws::SNS::Model::CreateTopicRequest request;
    request.SetName(topicName);

    const Aws::SNS::Model::CreateTopicOutcome outcome =
snsClient.CreateTopic(request);

    if (outcome.IsSuccess()) {
        topicARNResult = outcome.GetResult().GetTopicArn();
        std::cout << "Successfully created an Amazon SNS topic " << topicName
                  << " with topic ARN '" << topicARNResult
                  << "'." << std::endl;
    }
    else {
        std::cerr << "Error creating topic " << topicName << ":" <<
                  outcome.GetError().GetMessage() << std::endl;
        topicARNResult.clear();
    }

    return outcome.IsSuccess();
}

```

- Para obter detalhes da API, consulte [CreateTopic](#) Referência AWS SDK for C++ da API.

## CLI

### AWS CLI

Para criar um tópico do SNS

O exemplo `create-topic` a seguir cria um tópico do SNS chamado `my-topic`.

```
aws sns create-topic \  
  --name my-topic
```

Saída:

```
{  
  "ResponseMetadata": {  
    "RequestId": "1469e8d7-1642-564e-b85d-a19b4b341f83"  
  },  
  "TopicArn": "arn:aws:sns:us-west-2:123456789012:my-topic"  
}
```

Para obter mais informações, consulte [Usando a interface de linha de AWS comando com o Amazon SQS e o Amazon SNS](#) no Guia do usuário AWS da interface de linha de comando.

- Para obter detalhes da API, consulte [CreateTopic](#) na Referência de AWS CLI Comandos.

Go

SDK for Go V2

 Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```
// SnsActions encapsulates the Amazon Simple Notification Service (Amazon SNS)  
actions  
// used in the examples.  
type SnsActions struct {  
  SnsClient *sns.Client  
}  
  
// CreateTopic creates an Amazon SNS topic with the specified name. You can  
optionally  
// specify that the topic is created as a FIFO topic and whether it uses content-  
based
```

```
// deduplication instead of ID-based deduplication.
func (actor SnsActions) CreateTopic(topicName string, isFifoTopic bool,
contentBasedDeduplication bool) (string, error) {
    var topicArn string
    topicAttributes := map[string]string{}
    if isFifoTopic {
        topicAttributes["FifoTopic"] = "true"
    }
    if contentBasedDeduplication {
        topicAttributes["ContentBasedDeduplication"] = "true"
    }
    topic, err := actor.SnsClient.CreateTopic(context.TODO(), &sns.CreateTopicInput{
        Name:      aws.String(topicName),
        Attributes: topicAttributes,
    })
    if err != nil {
        log.Printf("Couldn't create topic %v. Here's why: %v\n", topicName, err)
    } else {
        topicArn = *topic.TopicArn
    }

    return topicArn, err
}
```

- Para obter detalhes da API, consulte [CreateTopic](#) Referência AWS SDK for Go da API.

## Java

### SDK para Java 2.x

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.sns.SnsClient;
import software.amazon.awssdk.services.sns.model.CreateTopicRequest;
import software.amazon.awssdk.services.sns.model.CreateTopicResponse;
```

```
import software.amazon.awssdk.services.sns.model.SnsException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */
public class CreateTopic {
    public static void main(String[] args) {
        final String usage = ""

            Usage:    <topicName>

            Where:
                topicName - The name of the topic to create (for example,
mytopic).

            """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String topicName = args[0];
        System.out.println("Creating a topic with name: " + topicName);
        SnsClient snsClient = SnsClient.builder()
            .region(Region.US_EAST_1)
            .build();

        String arnVal = createSNSTopic(snsClient, topicName);
        System.out.println("The topic ARN is" + arnVal);
        snsClient.close();
    }

    public static String createSNSTopic(SnsClient snsClient, String topicName) {
        CreateTopicResponse result;
        try {
            CreateTopicRequest request = CreateTopicRequest.builder()
                .name(topicName)

```

```
        .build();

        result = snsClient.createTopic(request);
        return result.topicArn();

    } catch (SnsException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
    return "";
}
}
```

- Para obter detalhes da API, consulte [CreateTopic](#) Referência AWS SDK for Java 2.x da API.

## JavaScript

### SDK para JavaScript (v3)

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

Crie o cliente em um módulo separado e exporte-o.

```
import { SNSClient } from "@aws-sdk/client-sns";

// The AWS Region can be provided here using the `region` property. If you leave
// it blank
// the SDK will default to the region set in your AWS config.
export const snsClient = new SNSClient({});
```

Importe o SDK e os módulos do cliente e chame a API.

```
import { CreateTopicCommand } from "@aws-sdk/client-sns";
import { snsClient } from "../libs/snsClient.js";
```

```
/**
 * @param {string} topicName - The name of the topic to create.
 */
export const createTopic = async (topicName = "TOPIC_NAME") => {
  const response = await snsClient.send(
    new CreateTopicCommand({ Name: topicName }),
  );
  console.log(response);
  // {
  //   '$metadata': {
  //     httpStatusCode: 200,
  //     requestId: '087b8ad2-4593-50c4-a496-d7e90b82cf3e',
  //     extendedRequestId: undefined,
  //     cfId: undefined,
  //     attempts: 1,
  //     totalRetryDelay: 0
  //   },
  //   TopicArn: 'arn:aws:sns:us-east-1:xxxxxxxxxxxx:TOPIC_NAME'
  // }
  return response;
};
```

- Para obter mais informações, consulte o [Guia do desenvolvedor do AWS SDK for JavaScript](#).
- Para obter detalhes da API, consulte [CreateTopic](#) Referência AWS SDK for JavaScript da API.

## Kotlin

### SDK for Kotlin

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```
suspend fun createSNSTopic(topicName: String): String {
```

```
val request = CreateTopicRequest {
    name = topicName
}

SnsClient { region = "us-east-1" }.use { snsClient ->
    val result = snsClient.createTopic(request)
    return result.topicArn.toString()
}
}
```

- Para obter detalhes da API, consulte a [CreateTopic](#) referência da API AWS SDK for Kotlin.

## PHP

### SDK para PHP

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
require 'vendor/autoload.php';

use Aws\Exception\AwsException;
use Aws\Sns\SnsClient;

/**
 * Create a Simple Notification Service topics in your AWS account at the
 * requested region.
 *
 * This code expects that you have AWS credentials set up per:
 * https://docs.aws.amazon.com/sdk-for-php/v3/developer-guide/
 * guide_credentials.html
 */

$SnsClient = new SnsClient([
    'profile' => 'default',
    'region' => 'us-east-1',
    'version' => '2010-03-31'
```

```
]);

$topicname = 'myTopic';

try {
    $result = $SnSClient->createTopic([
        'Name' => $topicname,
    ]);
    var_dump($result);
} catch (AwsException $e) {
    // output error message if fails
    error_log($e->getMessage());
}
```

- Para obter mais informações, consulte o [Guia do desenvolvedor do AWS SDK for PHP](#).
- Para obter detalhes da API, consulte [CreateTopic](#) Referência AWS SDK for PHP da API.

## Python

### SDK para Python (Boto3)

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```
class SnsWrapper:
    """Encapsulates Amazon SNS topic and subscription functions."""

    def __init__(self, sns_resource):
        """
        :param sns_resource: A Boto3 Amazon SNS resource.
        """
        self.sns_resource = sns_resource

    def create_topic(self, name):
        """
```

```
Creates a notification topic.

:param name: The name of the topic to create.
:return: The newly created topic.
"""
try:
    topic = self.sns_resource.create_topic(Name=name)
    logger.info("Created topic %s with ARN %s.", name, topic.arn)
except ClientError:
    logger.exception("Couldn't create topic %s.", name)
    raise
else:
    return topic
```

- Para obter detalhes da API, consulte a [CreateTopic](#) Referência da API AWS SDK for Python (Boto3).

## Ruby

### SDK for Ruby

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
# This class demonstrates how to create an Amazon Simple Notification Service
(SNS) topic.
class SNSTopicCreator
  # Initializes an SNS client.
  #
  # Utilizes the default AWS configuration for region and credentials.
  def initialize
    @sns_client = Aws::SNS::Client.new
  end

  # Attempts to create an SNS topic with the specified name.
  #
```

```

# @param topic_name [String] The name of the SNS topic to create.
# @return [Boolean] true if the topic was successfully created, false
otherwise.
def create_topic(topic_name)
  @sns_client.create_topic(name: topic_name)
  puts "The topic '#{topic_name}' was successfully created."
  true
rescue Aws::SNS::Errors::ServiceError => e
  # Handles SNS service errors gracefully.
  puts "Error while creating the topic named '#{topic_name}': #{e.message}"
  false
end
end

# Example usage:
if $PROGRAM_NAME == __FILE__
  topic_name = "YourTopicName" # Replace with your topic name
  sns_topic_creator = SNSTopicCreator.new

  puts "Creating the topic '#{topic_name}'..."
  unless sns_topic_creator.create_topic(topic_name)
    puts "The topic was not created. Stopping program."
    exit 1
  end
end
end

```

- Para obter mais informações, consulte o [Guia do desenvolvedor do AWS SDK for Ruby](#).
- Para obter detalhes da API, consulte [CreateTopic](#) Referência AWS SDK for Ruby da API.

## Rust

### SDK for Rust

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```

async fn make_topic(client: &Client, topic_name: &str) -> Result<(), Error> {

```

```
let resp = client.create_topic().name(topic_name).send().await?;

println!(
    "Created topic with ARN: {}",
    resp.topic_arn().unwrap_or_default()
);

Ok(())
}
```

- Para obter detalhes da API, consulte a [CreateTopic](#) referência da API AWS SDK for Rust.

## SAP ABAP

### SDK para SAP ABAP

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```
TRY.
    oo_result = lo_sns->createtopic( iv_name = iv_topic_name ). " oo_result
is returned for testing purposes. "
    MESSAGE 'SNS topic created' TYPE 'I'.
    CATCH /aws1/cx_snstopiclimitexcex.
    MESSAGE 'Unable to create more topics. You have reached the maximum
number of topics allowed.' TYPE 'E'.
ENDTRY.
```

- Para obter detalhes da API, consulte a [CreateTopic](#) referência da API AWS SDK for SAP ABAP.

# Assinar tópico do Amazon SNS

Para receber mensagens publicadas em um [tópico](#), você precisa inscrever um [endpoint](#) nesse tópico. Depois de inscrito, o endpoint começa a receber todas as mensagens publicadas no tópico associado.

## Note

Endpoints HTTP(S), endereços de e-mail e recursos da AWS em outras Contas da AWS exigem a confirmação da assinatura antes que possam receber mensagens.

## Como inscrever um endpoint em um tópico do Amazon SNS

1. Faça login no [console do Amazon SNS](#).
2. No painel de navegação à esquerda, escolha Assinaturas.
3. Na página Subscriptions (Assinaturas), escolha Create subscription (Criar assinatura).
4. Na página Create subscription (Criar inscrição), na seção Details (detalhes), faça o seguinte:
  - a. Em Topic ARN (ARN do tópico), escolha o nome do recurso da Amazon (ARN) de um tópico. Esse valor é o ARN da AWS que foi gerado quando você criou o tópico do Amazon SNS; por exemplo, `arn:aws:sns:us-east-2:123456789012:your_topic`.
  - b. Em Protocol (Protocolo), escolha um tipo de endpoint. Os tipos de endpoint disponíveis são:
    - [HTTP/HTTPS](#)
    - [Email/Email-JSON](#)
    - [Amazon Data Firehose](#)
    - [Amazon SQS](#)

## Note

Para se inscrever em um [tópico FIFO do SNS](#), escolha esta opção.

- [AWS Lambda](#)
- [Endpoint da aplicação da plataforma](#)

- [SMS](#)
- c. Em Endpoint, insira o valor do endpoint, como um endereço de e-mail ou o ARN de uma fila do Amazon SQS.
- d. Somente endpoints do Firehose: para o ARN da função de assinatura, especifique o ARN da função do IAM que você criou para gravar nos fluxos de entrega do Firehose. Para ter mais informações, consulte [Pré-requisitos para assinar streams de entrega do Firehose para tópicos do Amazon SNS](#).
- e. (Opcional) Para endpoints Firehose, Amazon SQS e HTTP/S, você também pode ativar a entrega de mensagens brutas. Para ter mais informações, consulte [Entrega de mensagens brutas do Amazon SNS](#).
- f. (Opcional) Para configurar uma política de filtros, expanda a seção Subscription filter policy (Política de filtro de assinatura). Para ter mais informações, consulte [Políticas de filtro de assinatura do Amazon SNS](#).
- g. (Opcional) Para habilitar a filtragem baseada em carga útil, configure Filter Policy Scope como MessageBody. Para ter mais informações, consulte [Escopo de política de filtro de assinaturas do Amazon SNS](#).
- h. (Opcional) Para configurar uma fila de mensagens mortas para a assinatura, expanda a seção Redrive policy (dead-letter queue) (Política de redirecionamento (fila de mensagens mortas)). Para ter mais informações, consulte [Filas de mensagens mortas \(DLQs\) do Amazon SNS](#).
- i. Selecione Criar assinatura.

O console cria a assinatura e abre a página Details (Detalhes) da assinatura.

## Excluindo um tópico e uma assinatura do Amazon SNS

Quando um tópico é excluído, suas assinaturas associadas são excluídas de forma assíncrona. Embora os clientes ainda possam acessar essas assinaturas, elas não estão mais associadas ao tópico, mesmo que você recrie o tópico usando o mesmo nome.

Se um assinante tentar publicar uma mensagem no tópico excluído, o editor receberá uma mensagem de erro indicando que o tópico não existe. Da mesma forma, qualquer tentativa de se inscrever no tópico excluído também resultará em uma mensagem de erro.

Não é possível excluir uma assinatura que esteja pendente de confirmação. Após 48 horas, o Amazon SNS exclui automaticamente a assinatura não confirmada.

## Tópicos

- [Para excluir um tópico ou uma assinatura do Amazon SNS usando o AWS Management Console](#)
- [Para excluir uma assinatura e um tópico usando o AWS SDK](#)

## Para excluir um tópico ou uma assinatura do Amazon SNS usando o AWS Management Console

Para excluir um tópico usando o AWS Management Console

1. Faça login no console [do Amazon SNS](#).
2. No painel de navegação à esquerda, selecione Tópicos.
3. Na página Topics (Tópicos), selecione um tópico e, em seguida, escolha Delete (Excluir).
4. Na caixa de diálogo Delete topic (Excluir tópico), insira delete me e escolha Delete (Excluir).

O console exclui o tópico.

Para excluir uma assinatura usando o AWS Management Console

1. Faça login no [console do Amazon SNS](#).
2. No painel de navegação à esquerda, escolha Assinaturas.
3. Na página Assinaturas, selecione uma assinatura com o status Confirmado e escolha Excluir.
4. Na caixa de diálogo Delete subscription (Excluir assinatura), escolha Delete (Excluir).

O console exclui a assinatura.

## Para excluir uma assinatura e um tópico usando o AWS SDK

Para usar um AWS SDK, você deve configurá-lo com suas credenciais. Para obter mais informações, consulte [The shared config and credentials files](#) (“Os arquivos compartilhados de configuração e de credenciais”) no Guia de referência de AWS SDKs e ferramentas.

Os exemplos de código a seguir mostram como usar `DeleteTopic`.

## .NET

### AWS SDK for .NET

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

Exclua um tópico por meio do respectivo ARN.

```
/// <summary>
/// Delete a topic by its topic ARN.
/// </summary>
/// <param name="topicArn">The ARN of the topic.</param>
/// <returns>True if successful.</returns>
public async Task<bool> DeleteTopicByArn(string topicArn)
{
    var deleteResponse = await _amazonSNSClient.DeleteTopicAsync(
        new DeleteTopicRequest()
        {
            TopicArn = topicArn
        });
    return deleteResponse.HttpStatusCode == HttpStatusCode.OK;
}
```

- Para obter detalhes da API, consulte [DeleteTópico](#) Referência AWS SDK for .NET da API.

## C++

### SDK para C++

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```
//! Delete an Amazon Simple Notification Service (Amazon SNS) topic.
/*!
  \param topicARN: The Amazon Resource Name (ARN) for an Amazon SNS topic.
  \param clientConfiguration: AWS client configuration.
  \return bool: Function succeeded.
*/
bool AwsDoc::SNS::deleteTopic(const Aws::String &topicARN,
                              const Aws::Client::ClientConfiguration
&clientConfiguration) {
    Aws::SNS::SNSClient snsClient(clientConfiguration);

    Aws::SNS::Model::DeleteTopicRequest request;
    request.SetTopicArn(topicARN);

    const Aws::SNS::Model::DeleteTopicOutcome outcome =
snsClient.DeleteTopic(request);

    if (outcome.IsSuccess()) {
        std::cout << "Successfully deleted the Amazon SNS topic " << topicARN <<
std::endl;
    }
    else {
        std::cerr << "Error deleting topic " << topicARN << ":" <<
outcome.GetError().GetMessage() << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Para obter detalhes da API, consulte [DeleteTopic](#) Referência AWS SDK for C++ da API.

## CLI

### AWS CLI

Para excluir um tópico do SNS

O exemplo `delete-topic` a seguir exclui o tópico do SNS especificado.

```
aws sns delete-topic \
  --topic-arn "arn:aws:sns:us-west-2:123456789012:my-topic"
```

Este comando não produz saída.

- Para obter detalhes da API, consulte [DeleteTopic](#) em Referência de AWS CLI Comandos.

Go

SDK for Go V2

 Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```
// SnsActions encapsulates the Amazon Simple Notification Service (Amazon SNS)
// actions
// used in the examples.
type SnsActions struct {
    SnsClient *sns.Client
}

// DeleteTopic delete an Amazon SNS topic.
func (actor SnsActions) DeleteTopic(topicArn string) error {
    _, err := actor.SnsClient.DeleteTopic(context.TODO(), &sns.DeleteTopicInput{
        TopicArn: aws.String(topicArn)})
    if err != nil {
        log.Printf("Couldn't delete topic %v. Here's why: %v\n", topicArn, err)
    }
    return err
}
```

- Para obter detalhes da API, consulte [DeleteTopic](#) em Referência AWS SDK for Go da API.

## Java

## SDK para Java 2.x

 Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.sns.SnsClient;
import software.amazon.awssdk.services.sns.model.DeleteTopicRequest;
import software.amazon.awssdk.services.sns.model.DeleteTopicResponse;
import software.amazon.awssdk.services.sns.model.SnsException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
 * started.html
 */
public class DeleteTopic {
    public static void main(String[] args) {
        final String usage = ""

                Usage:      <topicArn>

                Where:
                    topicArn - The ARN of the topic to delete.
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String topicArn = args[0];
        SnsClient snsClient = SnsClient.builder()
```

```
        .region(Region.US_EAST_1)
        .build();

        System.out.println("Deleting a topic with name: " + topicArn);
        deleteSNSTopic(snsClient, topicArn);
        snsClient.close();
    }

    public static void deleteSNSTopic(SnsClient snsClient, String topicArn) {
        try {
            DeleteTopicRequest request = DeleteTopicRequest.builder()
                .topicArn(topicArn)
                .build();

            DeleteTopicResponse result = snsClient.deleteTopic(request);
            System.out.println("\n\nStatus was " +
                result.sdkHttpResponse().statusCode());

        } catch (SnsException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Para obter detalhes da API, consulte [DeleteTopica](#) Referência AWS SDK for Java 2.x da API.

## JavaScript

### SDK para JavaScript (v3)

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

Crie o cliente em um módulo separado e exporte-o.

```
import { SNSClient } from "@aws-sdk/client-sns";
```

```
// The AWS Region can be provided here using the `region` property. If you leave
it blank
// the SDK will default to the region set in your AWS config.
export const snsClient = new SNSClient({});
```

Importe o SDK e os módulos do cliente e chame a API.

```
import { DeleteTopicCommand } from "@aws-sdk/client-sns";
import { snsClient } from "../libs/snsClient.js";

/**
 * @param {string} topicArn - The ARN of the topic to delete.
 */
export const deleteTopic = async (topicArn = "TOPIC_ARN") => {
  const response = await snsClient.send(
    new DeleteTopicCommand({ TopicArn: topicArn }),
  );
  console.log(response);
  // {
  //   '$metadata': {
  //     httpStatusCode: 200,
  //     requestId: 'a10e2886-5a8f-5114-af36-75bd39498332',
  //     extendedRequestId: undefined,
  //     cfId: undefined,
  //     attempts: 1,
  //     totalRetryDelay: 0
  //   }
  // }
};
```

- Para obter mais informações, consulte o [Guia do desenvolvedor do AWS SDK for JavaScript](#).
- Para obter detalhes da API, consulte [DeleteTopic](#) Referência AWS SDK for JavaScript da API.

## Kotlin

### SDK for Kotlin

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```
suspend fun deleteSNSTopic(topicArnVal: String) {  
  
    val request = DeleteTopicRequest {  
        topicArn = topicArnVal  
    }  
  
    SnsClient { region = "us-east-1" }.use { snsClient ->  
        snsClient.deleteTopic(request)  
        println("$topicArnVal was successfully deleted.")  
    }  
}
```

- Para obter detalhes da API, consulte a [DeleteTopic](#) referência da API AWS SDK for Kotlin.

## PHP

### SDK para PHP

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```
require 'vendor/autoload.php';  
  
use Aws\Exception\AwsException;  
use Aws\Sns\SnsClient;
```

```
/**
 * Deletes an SNS topic and all its subscriptions.
 *
 * This code expects that you have AWS credentials set up per:
 * https://docs.aws.amazon.com/sdk-for-php/v3/developer-guide/
guide_credentials.html
 */

$SnSClient = new SnsClient([
    'profile' => 'default',
    'region' => 'us-east-1',
    'version' => '2010-03-31'
]);

$topic = 'arn:aws:sns:us-east-1:111122223333:MyTopic';

try {
    $result = $SnSClient->deleteTopic([
        'TopicArn' => $topic,
    ]);
    var_dump($result);
} catch (AwsException $e) {
    // output error message if fails
    error_log($e->getMessage());
}
```

- Para obter detalhes da API, consulte [DeleteTopic](#) Referência AWS SDK for PHP da API.

## Python

SDK para Python (Boto3).

### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```
class SnsWrapper:
```

```
"""Encapsulates Amazon SNS topic and subscription functions."""

def __init__(self, sns_resource):
    """
    :param sns_resource: A Boto3 Amazon SNS resource.
    """
    self.sns_resource = sns_resource

    @staticmethod
    def delete_topic(topic):
        """
        Deletes a topic. All subscriptions to the topic are also deleted.
        """
        try:
            topic.delete()
            logger.info("Deleted topic %s.", topic.arn)
        except ClientError:
            logger.exception("Couldn't delete topic %s.", topic.arn)
            raise
```

- Para obter detalhes da API, consulte a [DeleteTopic](#) Referência da API AWS SDK for Python (Boto3).

## SAP ABAP

### SDK para SAP ABAP

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```
TRY.
  lo_sns->deletetopic( iv_topicarn = iv_topic_arn ).
  MESSAGE 'SNS topic deleted.' TYPE 'I'.
CATCH /aws1/cx_snsnotfoundexception.
  MESSAGE 'Topic does not exist.' TYPE 'E'.
```

```
ENDTRY.
```

- Para obter detalhes da API, consulte a [DeleteTopic](#) referência da API AWS SDK for SAP ABAP.

## Marcação de tópicos do Amazon SNS

O Amazon SNS oferece suporte à marcação de tópicos do Amazon SNS. Isso pode ajudar você a monitorar e gerenciar os custos associados aos seus tópicos, fornecer segurança reforçada nas suas políticas do AWS Identity and Access Management (IAM) e pesquisar ou filtrar facilmente milhares de tópicos. A marcação permite que você gerencie seus tópicos do Amazon SNS usando o AWS Resource Groups. Para obter mais informações sobre o Resource Groups, consulte o [Guia do usuário do AWS Resource Groups](#).

### Tópicos

- [Marcação para alocação de custos](#)
- [Marcação para controle de acesso](#)
- [Marcação para pesquisa e filtragem de recursos](#)
- [Configurar etiquetas de tópicos do Amazon SNS](#)

## Marcação para alocação de custos

Para organizar e identificar seus tópicos do Amazon SNS para alocação de custos, você pode adicionar etiquetas que identificam o objetivo de um tópico. Isso é especialmente útil quando você tem vários tópicos. Você pode usar tags de alocação de custos para organizar sua fatura da AWS para refletir sua própria estrutura de custos. Para isso, cadastre-se para obter a fatura da sua conta da AWS para incluir as chaves e valores das tags. Para obter mais informações, consulte [Configurar um relatório de alocação de custos mensal](#), no [Guia do usuário do AWS Billing and Cost Management](#).

Por exemplo, é possível adicionar etiquetas que representam o centro de custos e o objetivo dos seus tópicos do Amazon SNS, da seguinte maneira.

Recurso	Chave	Valor
Tópico 1	Centro de custos	43289
	Aplicação	Processamento de pedidos
Tópico 2	Centro de custos	43289
	Aplicação	Processamento de pagamentos
Tópico 3	Centro de custos	76585
	Aplicação	Arquivamento

Esse esquema de marcação permite que você agrupe dois tópicos executando tarefas relacionadas no mesmo centro de custos e, ao mesmo tempo, marcar uma atividade não relacionada com outra etiqueta de alocação de custos.

## Marcação para controle de acesso

O AWS Identity and Access Management oferece suporte para controlar o acesso a recursos com base em tags. Após a marcação dos seus recursos, forneça informações sobre as etiquetas de recurso no elemento de condição de uma política do IAM para gerenciar o acesso baseado em etiquetas. Para obter informações sobre como marcar seus recursos usando o [console do Amazon SNS](#) ou o [AWS SDK](#), consulte [Configurar etiquetas](#).

É possível restringir o acesso de uma identidade do IAM. Por exemplo, você pode restringir o acesso de Publish e PublishBatch a todos os tópicos do Amazon SNS que incluem uma etiqueta com a chave `environment` e o valor `production`, enquanto permite acesso a todos os outros tópicos do Amazon SNS. No exemplo abaixo, a política restringe a capacidade de publicar mensagens em tópicos marcados com `production`, enquanto permite que mensagens sejam publicadas em tópicos marcados com `development`. Para obter mais informações, consulte [Controlar o acesso usando etiquetas](#), no Guia do usuário do IAM.

### Note

Configurar a permissão do IAM para Publish define a permissão para Publish e PublishBatch.

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Deny",
    "Action": [
      "sns:Publish"
    ],
    "Resource": "arn:aws:sns:*:*:*",
    "Condition": {
      "StringEquals": {
        "aws:ResourceTag/environment": "production"
      }
    }
  },
  {
    "Effect": "Allow",
    "Action": [
      "sns:Publish"
    ],
    "Resource": "arn:aws:sns:*:*:*",
    "Condition": {
      "StringEquals": {
        "aws:ResourceTag/environment": "development"
      }
    }
  }
  ]
}
```

## Marcação para pesquisa e filtragem de recursos

Uma conta da AWS pode ter dezenas de milhares de tópicos do Amazon SNS (consulte [Cotas do Amazon SNS](#) para obter detalhes). Por meio da marcação de tópicos, é possível simplificar o processo de pesquisa ou filtragem de tópicos.

Por exemplo, você pode ter centenas de tópicos que estão associados ao seu ambiente de produção. Em vez de ter que pesquisar manualmente esses tópicos, você pode consultar todos eles com uma determinada etiqueta:

```
import com.amazonaws.services.resourcegroups.AWSResourceGroups;
import com.amazonaws.services.resourcegroups.AWSResourceGroupsClientBuilder;
import com.amazonaws.services.resourcegroups.model.QueryType;
import com.amazonaws.services.resourcegroups.model.ResourceQuery;
```

```
import com.amazonaws.services.resourcegroups.model.SearchResourcesRequest;
import com.amazonaws.services.resourcegroups.model.SearchResourcesResult;

public class Example {
    public static void main(String[] args) {
        // Query Amazon SNS Topics with tag "keyA" as "valueA"
        final String QUERY = "{\"ResourceTypeFilters\": [\"AWS::SNS::Topic\"],\
        \"TagFilters\": [{\"Key\": \"keyA\", \"Values\": [\"valueA\"]}]\"";

        // Initialize ResourceGroup client
        AWSResourceGroups awsResourceGroups = AWSResourceGroupsClientBuilder
            .standard()
            .build();

        // Query all resources with certain tags from ResourceGroups
        SearchResourcesResult result = awsResourceGroups.searchResources(
            new SearchResourcesRequest().withResourceQuery(
                new ResourceQuery()
                    .withType(QueryType.TAG_FILTERS_1_0)
                    .withQuery(QUERY)
            ));
        System.out.println("SNS Topics with certain tags are " +
            result.getResourceIdentifiers());
    }
}
```

## Configurar etiquetas de tópicos do Amazon SNS

Esta página mostra como você pode usar o AWS Management Console, um AWS SDK e a AWS CLI para configurar tags para um tópico do Amazon [SNS](#).

### Important

Não adicione informações de identificação pessoal (PII) nem outras informações confidenciais ou sigilosas em tags. As etiquetas são acessíveis a outros Amazon Web Services, incluindo o faturamento. As tags não devem ser usadas para dados privados ou confidenciais.

## Tópicos

- [Listar, adicionar e remover tags para um tópico do Amazon SNS usando o AWS Management Console](#)
- [Adicionar etiquetas a um tópico usando um AWS SDK](#)
- [Gerenciar etiquetas com ações de API do Amazon SNS](#)
- [Ações de API com suporte para o ABAC](#)

## Listar, adicionar e remover tags para um tópico do Amazon SNS usando o AWS Management Console

1. Faça login no [console do Amazon SNS](#).
2. No painel de navegação, escolha Topics (Tópicos).
3. Na página Topics (Tópicos), escolha um tópico e selecione Edit (Editar).
4. Expanda a seção Tags.

As tags adicionadas ao tópico serão listadas.

5. Modifique as tags do tópico:
  - Para adicionar uma etiqueta, escolha Add tag (Adicionar etiqueta) e insira uma Key (Chave) e um Value (Valor) (opcional).
  - Para remover uma tag, escolha Remove tag (Remover tag) ao lado de um par chave/valor.
6. Escolha Salvar alterações.

## Adicionar etiquetas a um tópico usando um AWS SDK

Para usar um AWS SDK, você deve configurá-lo com suas credenciais. Para obter mais informações, consulte [The shared config and credentials files](#) (“Os arquivos compartilhados de configuração e de credenciais”) no Guia de referência de AWS SDKs e ferramentas.

Os exemplos de código a seguir mostram como usar `TagResource`.

### CLI

#### AWS CLI

Para adicionar uma tag a um tópico

O exemplo `tag-resource` a seguir adiciona uma tag de metadados ao tópico do Amazon SNS especificado.

```
aws sns tag-resource \  
  --resource-arn arn:aws:sns:us-west-2:123456789012:MyTopic \  
  --tags Key=Team,Value=Alpha
```

Este comando não produz saída.

- Para obter detalhes da API, consulte [TagResource](#) na Referência de AWS CLI Comandos.

## Java

### SDK para Java 2.x

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```
import software.amazon.awssdk.regions.Region;  
import software.amazon.awssdk.services.sns.SnsClient;  
import software.amazon.awssdk.services.sns.model.SnsException;  
import software.amazon.awssdk.services.sns.model.Tag;  
import software.amazon.awssdk.services.sns.model.TagResourceRequest;  
import java.util.ArrayList;  
import java.util.List;  
  
/**  
 * Before running this Java V2 code example, set up your development  
 * environment, including your credentials.  
 *  
 * For more information, see the following documentation topic:  
 *  
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-  
started.html  
 */  
public class AddTags {  
    public static void main(String[] args) {  
        final String usage = ""
```

```
Usage:    <topicArn>

Where:
    topicArn - The ARN of the topic to which tags are added.

""";

if (args.length != 1) {
    System.out.println(usage);
    System.exit(1);
}

String topicArn = args[0];
SnsClient snsClient = SnsClient.builder()
    .region(Region.US_EAST_1)
    .build();

addTopicTags(snsClient, topicArn);
snsClient.close();
}

public static void addTopicTags(SnsClient snsClient, String topicArn) {
    try {
        Tag tag = Tag.builder()
            .key("Team")
            .value("Development")
            .build();

        Tag tag2 = Tag.builder()
            .key("Environment")
            .value("Gamma")
            .build();

        List<Tag> tagList = new ArrayList<>();
        tagList.add(tag);
        tagList.add(tag2);

        TagResourceRequest tagResourceRequest = TagResourceRequest.builder()
            .resourceArn(topicArn)
            .tags(tagList)
            .build();

        snsClient.tagResource(tagResourceRequest);
        System.out.println("Tags have been added to " + topicArn);
    }
}
```

```
        } catch (SnsException e) {  
            System.err.println(e.awsErrorDetails().errorMessage());  
            System.exit(1);  
        }  
    }  
}
```

- Para obter detalhes da API, consulte [TagResource](#) Referência AWS SDK for Java 2.x da API.

## Kotlin

### SDK for Kotlin

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```
suspend fun addTopicTags(topicArn: String) {  
  
    val tag = Tag {  
        key = "Team"  
        value = "Development"  
    }  
  
    val tag2 = Tag {  
        key = "Environment"  
        value = "Gamma"  
    }  
  
    val tagList = mutableListOf<Tag>()  
    tagList.add(tag)  
    tagList.add(tag2)  
  
    val request = TagResourceRequest {  
        resourceArn = topicArn  
        tags = tagList  
    }  
}
```

```
    }  
  
    SnsClient { region = "us-east-1" }.use { snsClient ->  
        snsClient.tagResource(request)  
        println("Tags have been added to $topicArn")  
    }  
}
```

- Para obter detalhes da API, consulte a [TagResource](#) referência da API AWS SDK for Kotlin.

## Gerenciar etiquetas com ações de API do Amazon SNS

Para gerenciar etiquetas usando a API do Amazon SNS, use as seguintes ações de API:

- [ListTagsForResource](#)
- [TagResource](#)
- [UntagResource](#)

## Ações de API com suporte para o ABAC

A lista a seguir inclui as ações de API que oferecem suporte ao controle de acesso baseado em atributos (ABAC). Para obter mais detalhes sobre o ABAC, consulte Para [que serve o ABAC?](#) AWS no Guia do usuário do IAM.

- [AddPermission](#)
- [ConfirmSubscription](#)
- [DeleteTopic](#)
- [GetDataProtectionPolicy](#)
- [GetSubscriptionAttributes](#)
- [GetTopicAttributes](#)
- [ListSubscriptionsByTopic](#)
- [ListTagsForResource](#)
- [Publish](#)
- [PublishBatch](#)
- [PutDataProtectionPolicy](#)

- [RemovePermission](#)
- [SetSubscriptionAttributes](#)
- [SetTopicAttributes](#)
- [Subscribe](#)
- [TagResource](#)
- [Unsubscribe](#)
- [UntagResource](#)

# Ordenação de mensagens e desduplicação (tópicos FIFO)

Você pode usar tópicos FIFO (primeiro a entrar, primeiro a sair) do Amazon SNS com [filas FIFO do Amazon SQS](#) para fornecer ordenação estrita e desduplicação de mensagens. Os recursos FIFO de cada um desses serviços trabalham em conjunto para atuar como um serviço totalmente gerenciado para integrar aplicações distribuídas que exigem consistência de dados quase em tempo real. A inscrição de [filas padrão do Amazon SQS](#) em tópicos FIFO do Amazon SNS oferece a melhor ordenação possível e a entrega pelo menos uma vez.

## Tópicos

- [Exemplo de caso de uso de tópicos FIFO](#)
- [Detalhes de ordenação de mensagens para tópicos FIFO](#)
- [Agrupamento de mensagens para tópicos FIFO](#)
- [Entrega de mensagens para tópicos FIFO](#)
- [Filtragem de mensagens para tópicos FIFO](#)
- [Desduplicação de mensagens para tópicos FIFO](#)
- [Segurança de mensagens para tópicos FIFO](#)
- [Durabilidade de mensagens para tópicos FIFO](#)
- [Arquivamento e reprodução de mensagens de tópicos FIFO](#)
- [Exemplos de código para tópicos FIFO](#)

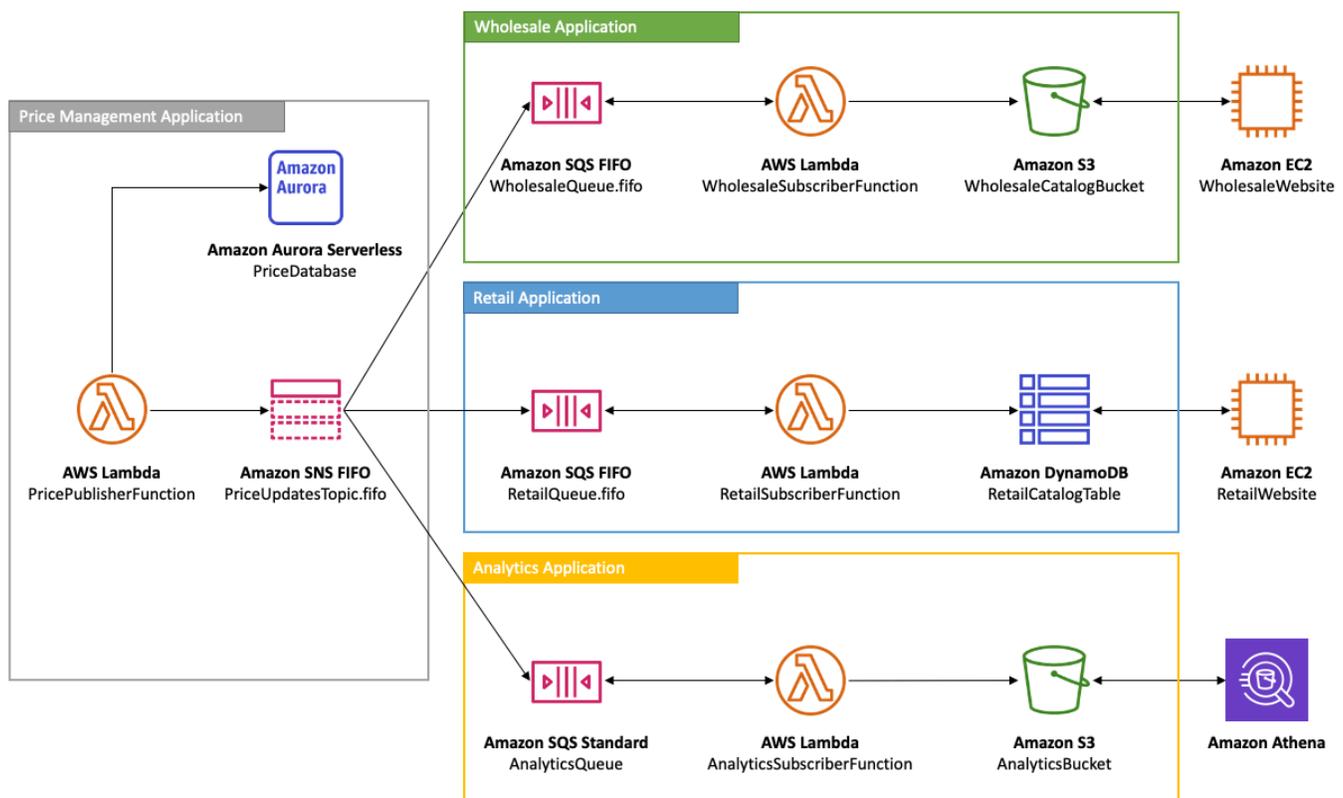
## Exemplo de caso de uso de tópicos FIFO

O exemplo a seguir descreve uma plataforma de comércio eletrônico criada por um fabricante de autopeças usando tópicos FIFO do Amazon SNS e filas do Amazon SQS. A plataforma é composta de quatro aplicações sem servidor:

- Os gerentes de inventário usam uma aplicação de gerenciamento de preços para definir o preço de cada item em estoque. Nessa empresa, os preços dos produtos podem mudar com base na flutuação cambial, na demanda do mercado e em mudanças na estratégia de vendas. A aplicação de gerenciamento de preços usa uma função do AWS Lambda que publica atualizações de preços em um tópico FIFO do Amazon SNS sempre que os preços mudam.
- Uma aplicação de atacado fornece o backend para um site em que oficinas mecânicas e fabricantes de automóveis podem comprar peças automotivas da empresa a granel. Para obter

notificações sobre alteração de preço, a aplicação de atacado inscreve sua fila FIFO do Amazon SQS no tópico FIFO do Amazon SNS da aplicação de gerenciamento de preços.

- Uma aplicação de varejo fornece o backend para outro site em que os proprietários de carros e entusiastas de carros tunados podem comprar peças automotivas individuais para seus veículos. Para obter notificações sobre alteração de preço, a aplicação de varejo também inscreve sua fila FIFO do Amazon SQS no tópico FIFO do Amazon SNS da aplicação de gerenciamento de preços.
- Uma aplicação de análise que agrega atualizações de preços e as armazena em um bucket do Amazon S3, permitindo que o Amazon Athena consulte o bucket para fins de business intelligence (BI). Para obter notificações sobre alteração de preço, a aplicação de análise inscreve sua fila padrão do Amazon SQS no tópico FIFO do Amazon SNS da aplicação de gerenciamento de preços. Ao contrário de outras aplicações, a de análise não exige que as atualizações de preços sejam ordenadas estritamente.

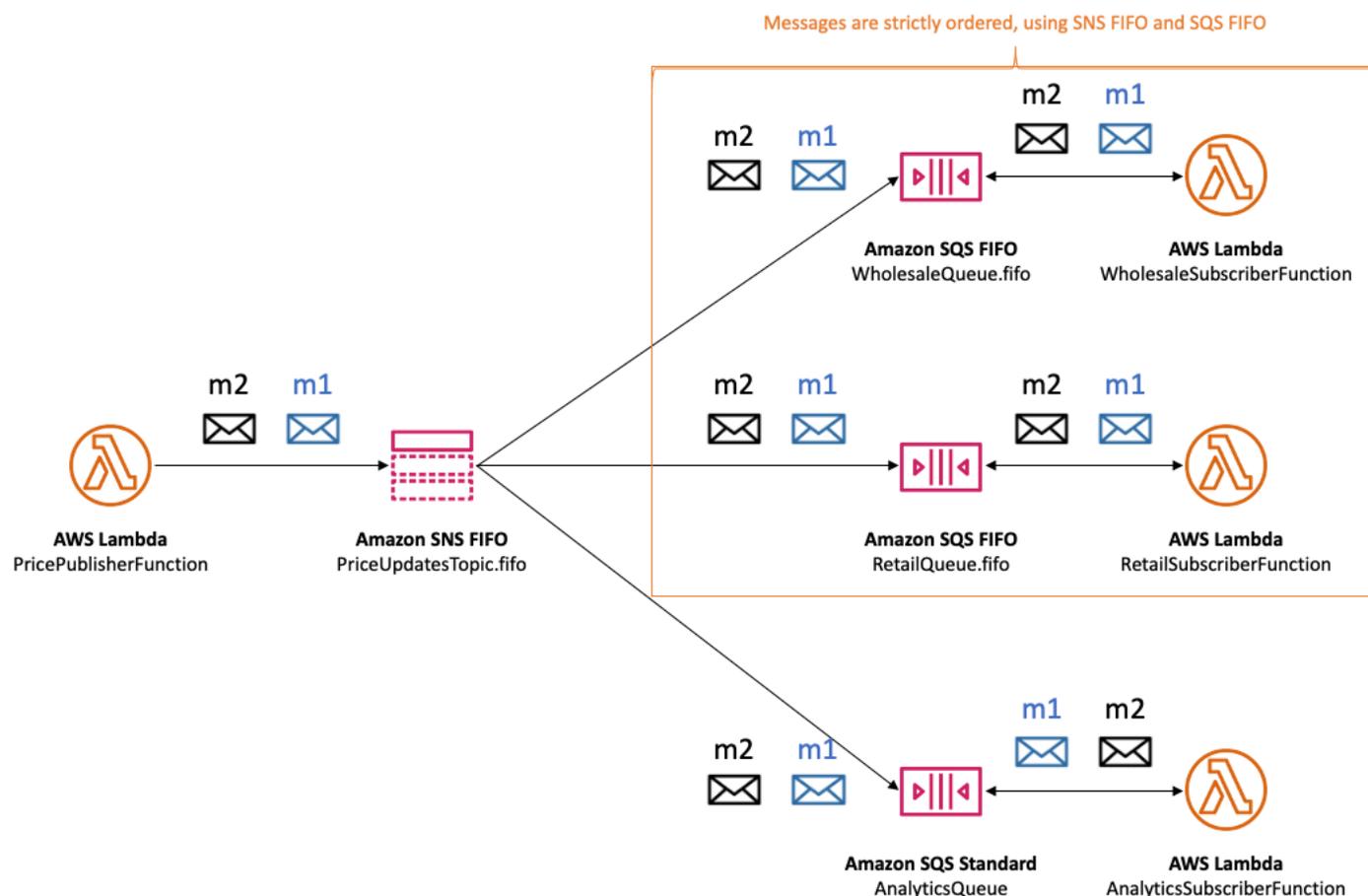


Para que as aplicações de atacado e varejo recebam atualizações de preços na ordem correta, a aplicação de gerenciamento de preços deve usar um sistema de distribuição de mensagens estritamente ordenado. O uso de tópicos FIFO do Amazon SNS e filas FIFO do Amazon SQS permite o processamento de mensagens em ordem e sem duplicação. Para obter mais informações,

consulte [Detalhes de ordenação de mensagens para tópicos FIFO](#). Para obter trechos de código que implementem esse caso de uso, consulte [Exemplos de código para tópicos FIFO](#).

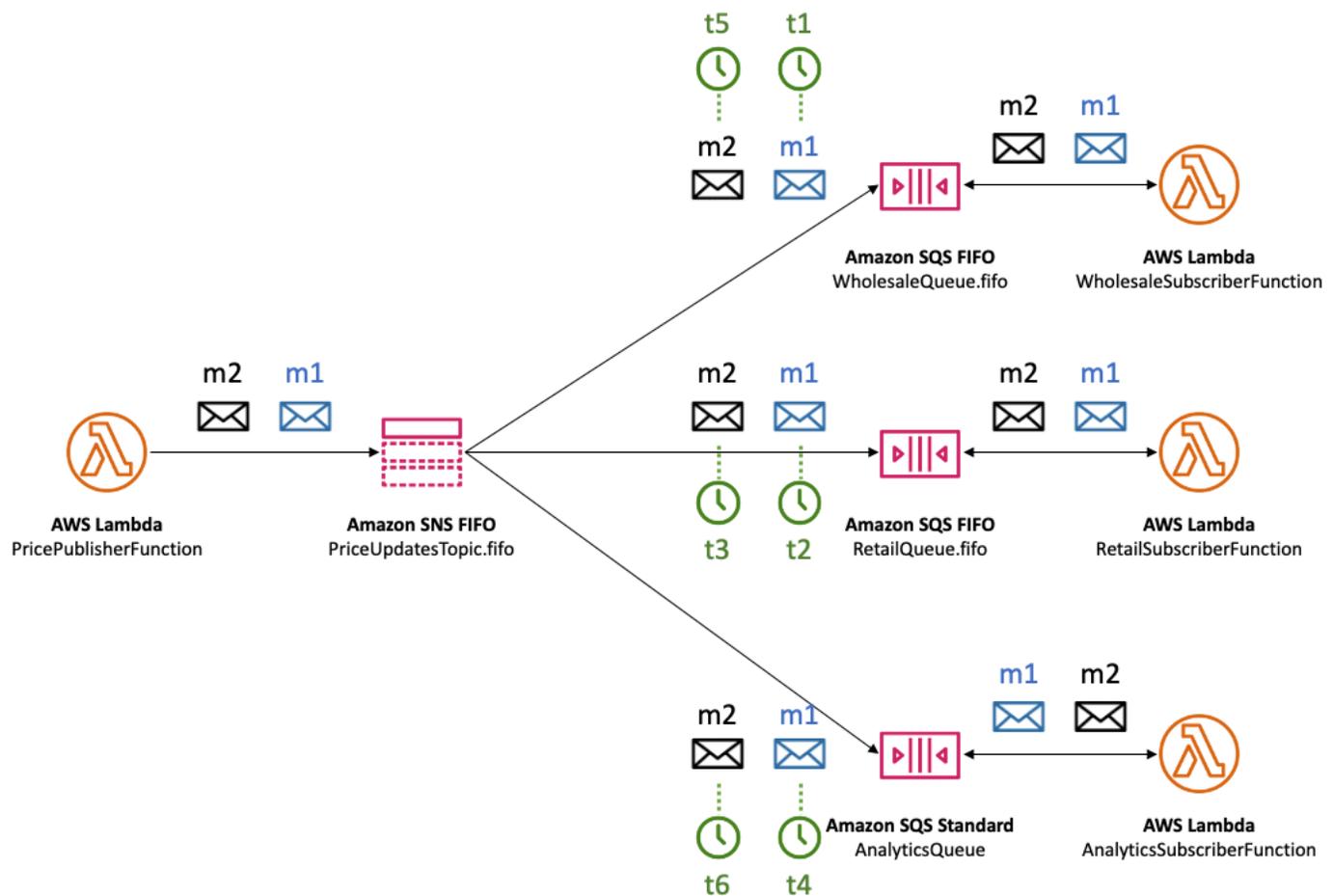
## Detalhes de ordenação de mensagens para tópicos FIFO

Um tópico FIFO do Amazon SNS sempre entrega mensagens para filas do Amazon SQS inscritas na ordem exata em que as mensagens são publicadas no tópico e somente uma vez. Com uma fila FIFO do Amazon SQS assinada, o consumidor da fila recebe as mensagens na ordem exata em que são entregues à fila, sem duplicatas. No entanto, com uma fila padrão do Amazon SQS assinada, o consumidor da fila pode receber mensagens fora de ordem e mais de uma vez. Isso permite uma maior dissociação entre assinantes e editores, oferecendo aos assinantes maior flexibilidade em termos de consumo de mensagens e otimização de custos, conforme mostrado no diagrama a seguir, com base no [Exemplo de caso de uso de tópicos FIFO](#).

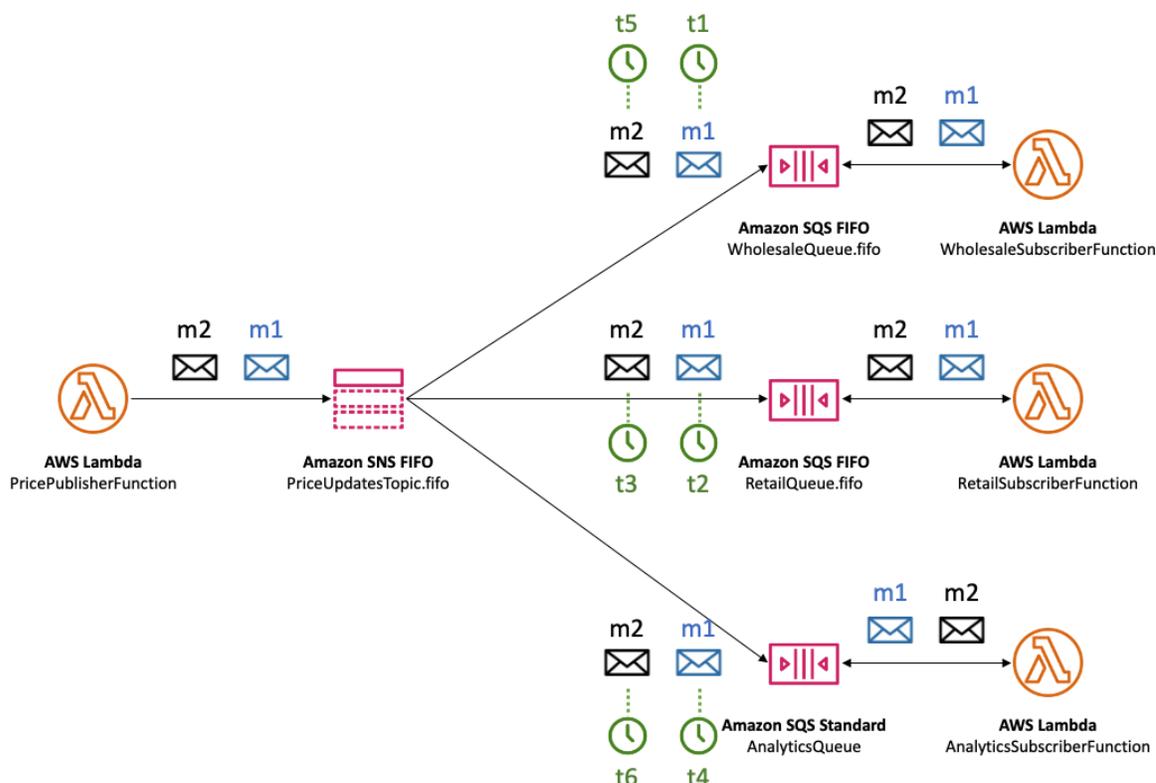


Observe que não há nenhuma ordenação implícita dos assinantes. O exemplo a seguir mostra que essa mensagem **m1** é entregue primeiro ao assinante atacadista e, depois, ao assinante varejista e, então ao assinante analista. A mensagem **m2** é entregue primeiro ao assinante varejista, depois, ao

assinante atacadista e, por fim, ao assinante analista. Embora as duas mensagens sejam entregues aos assinantes em uma ordem diferente, a sequência das mensagens é preservada para cada assinante de FIFO do Amazon SQS. Cada assinante é percebido isoladamente de qualquer outro assinante.

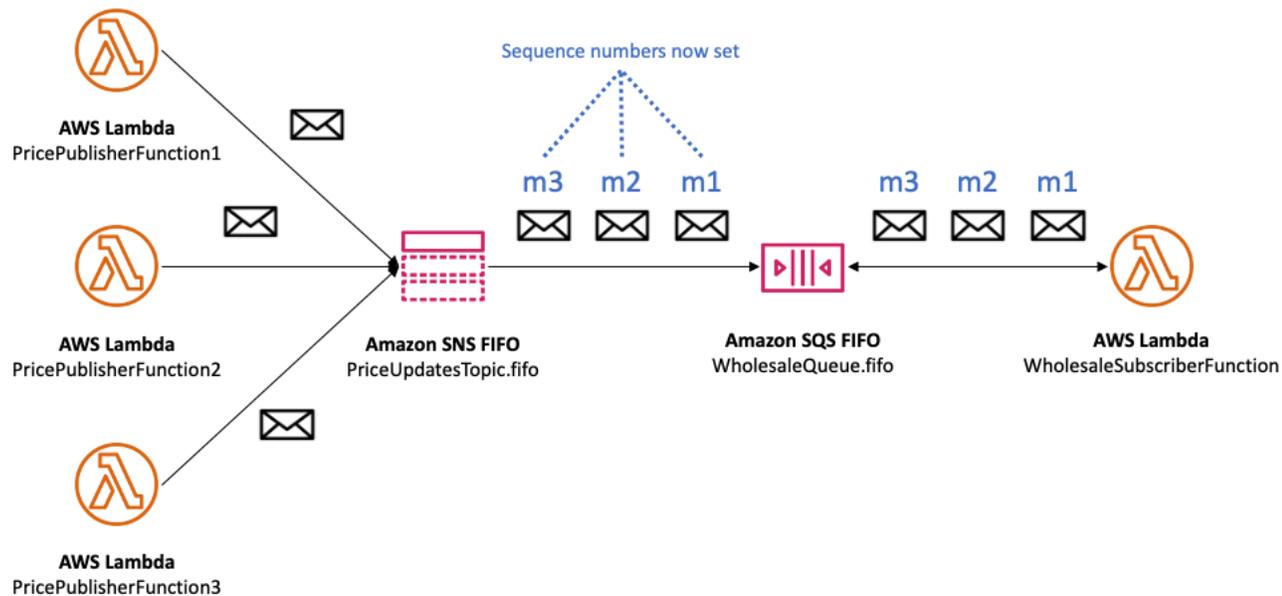


Se um assinante da fila do Amazon SQS se tornar inacessível, ele poderá ficar fora de sincronia. Por exemplo, digamos que o proprietário da fila da aplicação de atacado altere erroneamente a [política de fila do Amazon SQS](#) de uma maneira que impede que o principal de serviço do Amazon SNS entregue mensagens para a fila. Nesse caso, as entregas de atualização de preço à fila de atacado falham, enquanto as entregas às filas de varejo e análise são bem-sucedidas, fazendo com que os assinantes fiquem fora de sincronia. Quando o proprietário da fila da aplicação de atacado corrige a política de fila, o Amazon SNS retoma a entrega de mensagens para a fila inscrita. Todas as mensagens publicadas no tópico que se destinam à fila configurada incorretamente são descartadas, a menos que a assinatura tenha uma [fila de mensagens não entregues](#) configurada.



Você pode ter várias aplicações (ou várias threads dentro da mesma aplicação) publicando mensagens em um tópico FIFO do SNS em paralelo. Ao fazer isso, você efetivamente delega o sequenciamento de mensagens ao serviço Amazon SNS. Para determinar a sequência estabelecida de mensagens, você pode verificar o número de sequência.

O número de sequência é um número grande, não consecutivo que o Amazon SNS atribui a cada mensagem. O tamanho do número de sequência é de 128 bits e continua aumentando para cada [grupo de mensagens](#). O número de sequência é passado para as filas do Amazon SQS inscritas como parte do corpo da mensagem. No entanto, se você habilitar a [entrega de mensagens brutas](#), a mensagem entregue à fila do Amazon SQS não incluirá o número de sequência nem nenhum outro metadado de mensagens do Amazon SNS.



Os tópicos FIFO do Amazon SNS definem pedidos no contexto de um grupo de mensagens. Para obter mais informações, consulte [Agrupamento de mensagens para tópicos FIFO](#).

## Agrupamento de mensagens para tópicos FIFO

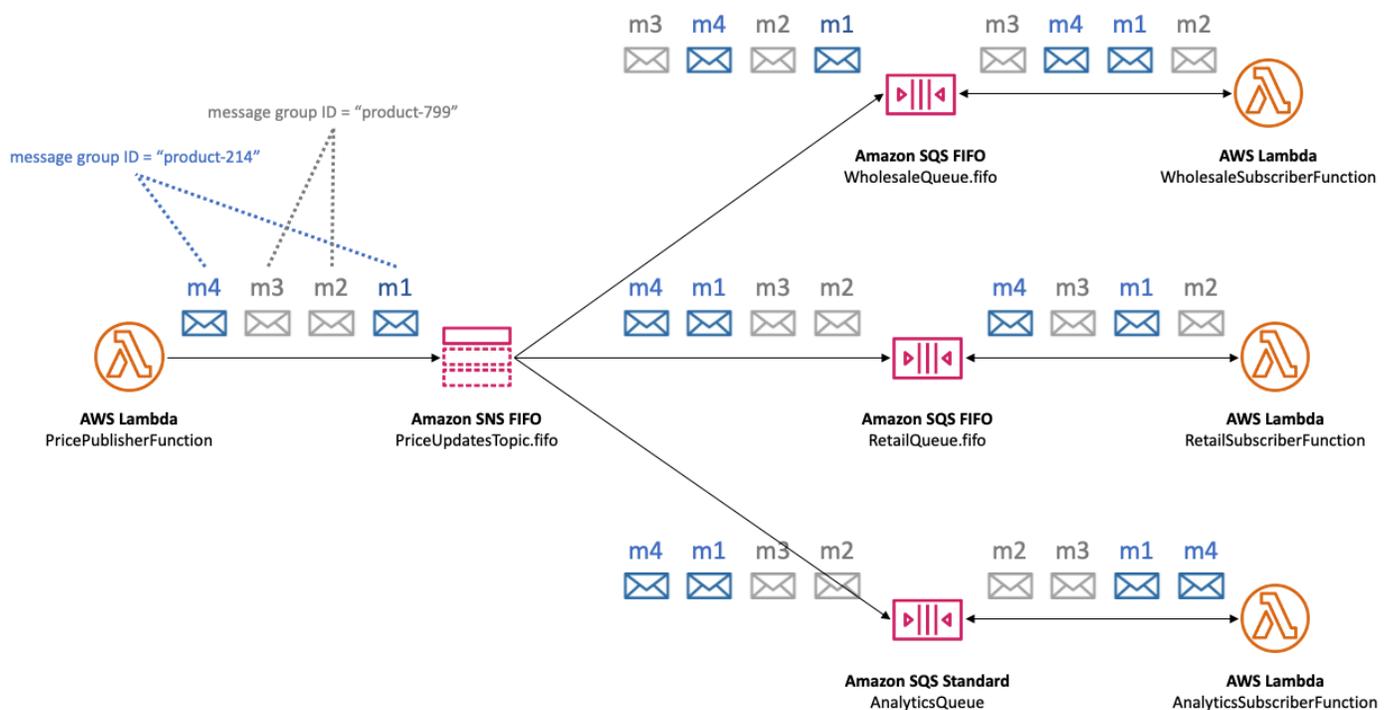
As mensagens que pertencem ao mesmo grupo são processadas uma a uma, em uma ordem estrita em relação ao grupo.

Ao publicar mensagens em um tópico FIFO do Amazon SNS, você define o ID do grupo de mensagens. O ID do grupo é um token obrigatório que determina que uma mensagem pertence a um grupo de mensagens específico. O tópico FIFO do SNS passa o ID do grupo para as filas FIFO inscritas do Amazon SQS. Não há limite para o número de IDs de grupo em tópicos FIFO do SNS ou filas FIFO do SQS. O ID do grupo de mensagens não é passado para filas padrão do Amazon SQS.

Não há afinidade entre um grupo de mensagens e uma assinatura. Portanto, as mensagens publicadas em qualquer grupo de mensagens são entregues a todas as filas inscritas, sujeitas a quaisquer políticas de filtro anexadas a assinaturas. Para obter mais informações, consulte [Entrega de mensagens para tópicos FIFO](#) e [Filtragem de mensagens para tópicos FIFO](#).

No [exemplo de caso de uso de gerenciamento de preços de peças automotivas](#), há um grupo de mensagens dedicado para cada produto vendido na plataforma. O mesmo tópico FIFO do Amazon SNS é usado para processar todas as atualizações de preços. A sequência de atualizações de preços é preservada no contexto de um único produto de peças automotivas, mas não em vários produtos. O diagrama a seguir mostra como isso funciona. Observe que, para o produto cujo ID de

grupo de mensagens é product-214, a mensagem m1 é processada antes da mensagem m4. Essa sequência é preservada em todos os fluxos de trabalho que usam FIFO do Amazon SNS a FIFO do Amazon SQS. Da mesma forma, para o produto cujo ID de grupo de mensagens é product-799, a mensagem m2 é processada antes da mensagem m3, desde que os fluxos de trabalho usem FIFO do Amazon SNS e FIFO do Amazon SQS. No entanto, ao usar as filas padrão do Amazon SQS, a ordem das mensagens não é mais garantida e os grupos de mensagens não existem. Os grupos de mensagens product-214 e product-799 são independentes entre si, portanto, não há relação entre a forma como as mensagens são sequenciadas.



## Distribuição de dados por IDs de grupo de mensagens para melhor desempenho

Para otimizar o throughput de entrega, os tópicos FIFO do Amazon SNS entregam mensagens de diferentes grupos de mensagens em paralelo, enquanto a ordem das mensagens é estritamente mantida dentro de cada grupo de mensagens. Cada grupo de mensagens individual pode entregar no máximo 300 mensagens por segundo. Portanto, para obter alto throughput para um único tópico, use um grande número de IDs de grupos de mensagens distintos. Ao utilizar um conjunto diversificado de grupos de mensagens, os tópicos FIFO do Amazon SNS distribuem automaticamente as mensagens em um número maior de partições paralelas.

**Note**

Os tópicos FIFO do Amazon SNS são otimizados para distribuição uniforme de mensagens entre IDs de grupos de mensagens, independentemente do número de grupos. A AWS recomenda que você use um grande número de IDs de grupos de mensagens distintas para otimizar o desempenho.

Ao publicar em seu tópico FIFO do Amazon SNS com alto throughput e uma ou mais filas FIFO do Amazon SQS estarem assinadas, é recomendável que você habilite o alto throughput nas filas. Para obter mais informações, consulte [Alta taxa de transferência para filas FIFO](#) no Guia do desenvolvedor do Amazon Simple Queue Service.

## Entrega de mensagens para tópicos FIFO

Os tópicos FIFO (primeiro a entrar, primeiro a sair) do Amazon SNS são compatíveis com a entrega para as filas padrão e FIFO do Amazon SQS para oferecer aos clientes flexibilidade e controle ao integrar aplicações distribuídas que exigem consistência de dados quase em tempo real.

Para workloads que precisam preservar a ordem estrita de mensagens ou a desduplicação, a combinação dos tópicos FIFO do Amazon SNS com [filas FIFO do Amazon SQS](#), inscritas como endpoint de entrega, aprimora o sistema de mensagens entre aplicações quando a ordem das operações e dos eventos é crucial ou quando não se admitem duplicatas.

Para workloads que comportam a melhor ordenação possível (e a entrega pelo menos uma vez, inscrever [filas padrão do Amazon SQS](#) nos tópicos FIFO do Amazon SNS permite reduzir custos, além de compartilhar filas entre workloads que não utilizam FIFO.

**Note**

Para exibir mensagens dos tópicos FIFO do Amazon SNS para funções do AWS Lambda, são necessárias etapas adicionais. Primeiro, inscreva as filas padrão ou FIFO do Amazon SQS no tópico. Em seguida, configure as filas para acionar as funções. Para obter mais informações, consulte [SQS FIFO as an event source](#) (“FIFO do SQS como fonte de eventos”) no Blog de computação da AWS.

Os tópicos FIFO do SNS não podem entregar mensagens para endpoints gerenciados pelo cliente, como endereços de e-mail, aplicativos móveis, números de telefone para sistemas de mensagens de texto (SMS) ou endpoints HTTP(S). Esses tipos de endpoint não têm garantia de preservar a ordenação estrita de mensagens. As tentativas de inscrever endpoints gerenciados pelo cliente nos tópicos FIFO do SNS resultam em erros.

Os tópicos FIFO do SNS oferecem suporte aos mesmos recursos de filtragem de mensagens que os tópicos padrão. Para obter mais informações, consulte [Filtragem de mensagens para tópicos FIFO](#) e [Simplify Your Pub/Sub Messaging with Amazon SNS Message Filtering](#) (“Simplificar seu sistema de mensagens Pub/Sub com a filtragem de mensagens do Amazon SNS”) no Blog de computação da AWS.

## Filtragem de mensagens para tópicos FIFO

Os tópicos FIFO do Amazon SNS oferecem suporte à filtragem de mensagens. O uso da filtragem de mensagens simplifica sua arquitetura descarregando a lógica de roteamento de mensagens dos sistemas do editor e a lógica de filtragem de mensagens de seus sistemas assinantes.

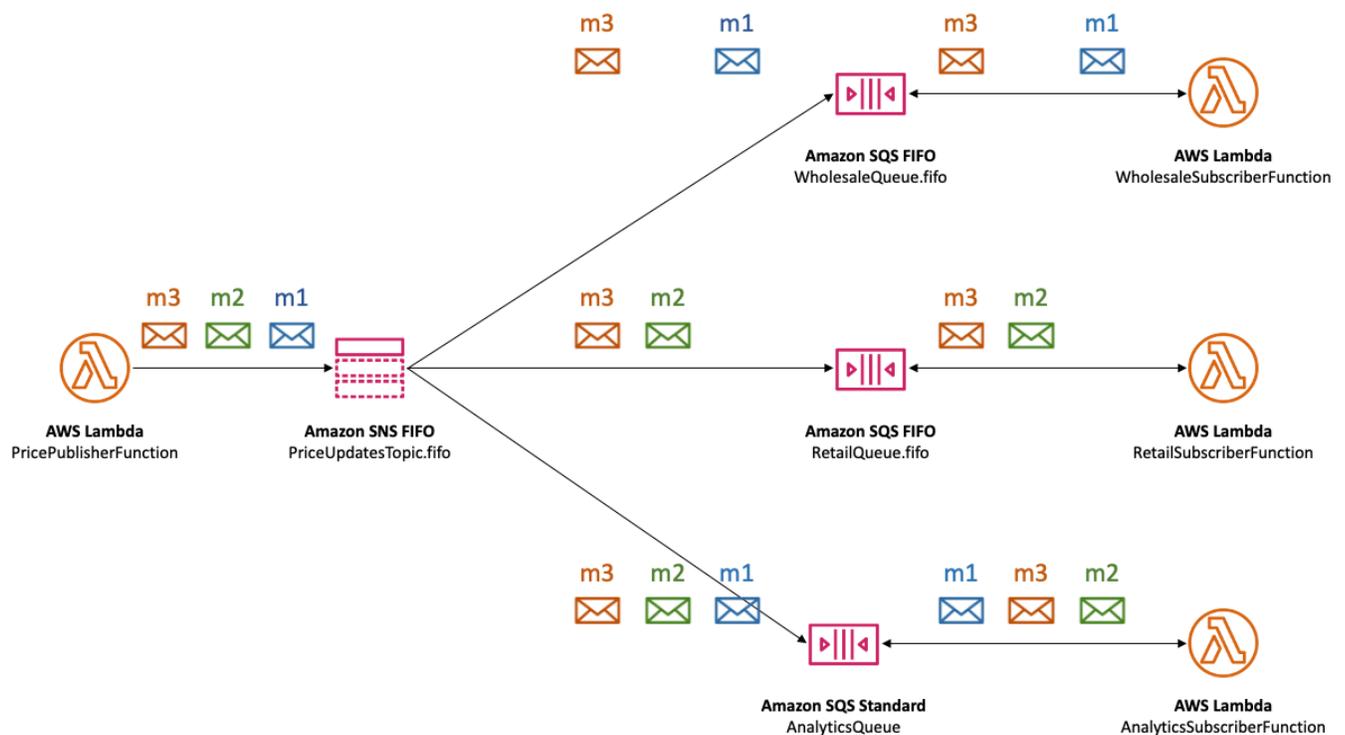
Ao inscrever uma fila FIFO ou padrão do Amazon SQS em um tópico FIFO do SNS, você pode usar a filtragem de mensagens para especificar que o assinante receba um subconjunto de mensagens, em vez de todas elas. Cada assinante pode definir sua própria política de filtro como atributos de assinatura. Com base no escopo da política de filtro, a política de filtro é comparada aos atributos ou ao corpo da mensagem recebida. Se a política de filtro estabelecer correspondência, o tópico entregará uma cópia da mensagem ao assinante. Se não houver correspondência, o tópico não entregará uma cópia da mensagem.

No [exemplo de caso de uso de gerenciamento de preços de peças automotivas](#), suponha que as seguintes políticas de filtro do Amazon SNS estejam definidas e que o escopo de política seja `MessageBody`:

- Para a fila de atacado, a política de filtro `{"business":["wholesale"]}` estabelece correspondência com cada mensagem que contém uma chave chamada `business` e com `wholesale` no conjunto de valores. No diagrama a seguir, uma das chaves na mensagem `m1` é `business` com um valor de `wholesale`. Uma das chaves na mensagem `m3` é `business` com um valor de `["wholesale,retail"]`. Assim, tanto `m1` quanto `m3` correspondem aos critérios da política de filtro e ambas as mensagens são entregues à fila de atacado.
- Para a fila de varejo, a política de filtro `{"business":["retail"]}` estabelece correspondência com cada mensagem que contém uma chave chamada `business` e com `retail` no conjunto de

valores. No diagrama, uma das chaves na mensagem m2 é `business` com um valor de `retail`. Uma das chaves na mensagem m3 é `business` com o valor de `["wholesale, retail"]`. Assim, tanto m2 quanto m3 correspondem aos critérios da política de filtro e ambas as mensagens são entregues à fila de varejo.

- Com relação à fila de análise, queremos que o Amazon Athena receba todos os registros, para que nenhuma política de filtro seja aplicada.



Os tópicos FIFO do SNS oferecem suporte a uma variedade de operadores correspondentes, incluindo valores de string de atributo, valores numéricos de atributo e chaves de atributo. Para obter mais informações, consulte [Filtragem de mensagens do Amazon SNS](#).

Os tópicos FIFO do SNS não entregam mensagens duplicadas para endpoints inscritos. Para obter mais informações, consulte [Desduplicação de mensagens para tópicos FIFO](#).

## Desduplicação de mensagens para tópicos FIFO

Os tópicos FIFO do Amazon SNS e as filas FIFO do Amazon SQS oferecem suporte à desduplicação de mensagens, que fornece entrega e processamento de mensagens exatamente uma vez, desde que as seguintes condições sejam atendidas:

- A fila FIFO do Amazon SQS inscrita existe e tem permissões para que a entidade principal de serviço do Amazon SNS entregue mensagens para a fila.
- O consumidor de fila FIFO do Amazon SQS processa a mensagem e a exclui da fila antes que o tempo limite de visibilidade expire.
- O tópico de assinatura do Amazon SNS não tem [filtragem de mensagens](#). Quando você configura a filtragem de mensagens, os tópicos FIFO do Amazon SNS at-most-once oferecem suporte à entrega, pois as mensagens podem ser filtradas com base nas suas políticas de filtro de assinatura.
- Não há interrupções de rede que impeçam a confirmação da entrega da mensagem.

 Note

A deduplicação de mensagens aplica-se a um tópico FIFO do Amazon SNS inteiro, não a um [grupo de mensagens](#) individual.

Quando você publica uma mensagem em um tópico FIFO do Amazon SNS, a mensagem deve incluir um ID de deduplicação. Esse ID está incluído na mensagem que o tópico FIFO do Amazon SNS entrega às filas FIFO do Amazon SQS inscritas.

Se uma mensagem com um ID de deduplicação específico for publicada com êxito em um tópico FIFO do SNS, qualquer mensagem publicada com o mesmo ID de deduplicação, dentro do intervalo de deduplicação de cinco minutos, será aceita, mas não entregue. O tópico FIFO do Amazon SNS continua a rastrear o ID de deduplicação da mensagem, mesmo depois que a mensagem é entregue aos endpoints inscritos.

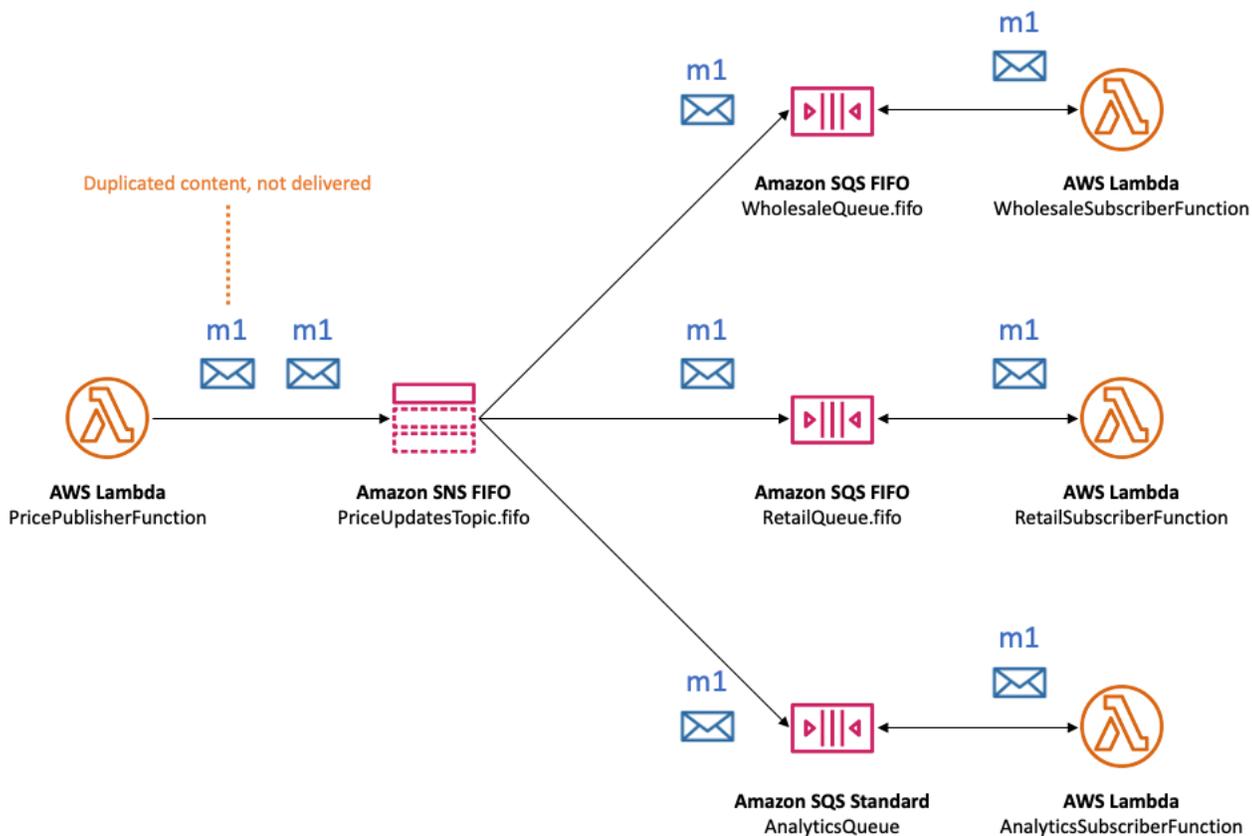
Se houver garantia de que o corpo da mensagem será exclusivo para cada mensagem publicada, você poderá habilitar a deduplicação baseada em conteúdo para um tópico FIFO do Amazon SNS e as filas FIFO do Amazon SQS inscritas. O Amazon SNS usa o corpo da mensagem para gerar um valor de hash exclusivo a ser usado como o ID de deduplicação para cada mensagem, portanto, você não precisa definir um ID de deduplicação ao enviar cada mensagem.

 Note

Os atributos de mensagem não estão incluídos no cálculo de hash.

Quando a deduplicação baseada em conteúdo é habilitada para um tópico FIFO do Amazon SNS e uma mensagem é publicada com uma ID de deduplicação, a ID de deduplicação publicada substitui a ID de deduplicação baseada em conteúdo gerada.

No [caso de uso de exemplo de gerenciamento de preços de peças automotivas](#), a empresa deve definir um ID de deduplicação universalmente exclusivo para cada atualização de preço. Isso ocorre porque o corpo da mensagem pode ser idêntico mesmo quando o atributo de mensagem é diferente para atacado e varejo. No entanto, se a empresa adicionasse o tipo de negócio (atacado ou varejo) ao corpo da mensagem com o ID e o preço do produto, ela poderia permitir a duplicação baseada em conteúdo no tópico FIFO do Amazon SNS e nas filas FIFO do Amazon SQS inscritas.



Além da ordenação e deduplicação de mensagens, os tópicos FIFO do Amazon SNS oferecem suporte à criptografia do lado do servidor (SSE) de mensagens com chaves e à privacidade de mensagens por meio de VPC endpoints AWS KMS com AWS PrivateLink. Para ter mais informações, consulte [Segurança de mensagens para tópicos FIFO](#).

## Segurança de mensagens para tópicos FIFO

Você pode optar por fazer com que o Amazon SNS e o Amazon SQS criptografem mensagens enviadas para tópicos e filas FIFO, usando [chaves mestras personalizadas \(CMKs\)](#) do [AWS Key Management Service \(AWS KMS\)](#). Você pode criar tópicos e filas FIFO criptografados ou optar por criptografar tópicos e filas FIFO existentes. O Amazon SNS e o Amazon SQS criptografam apenas o corpo da mensagem. Eles não criptografam os atributos da mensagem, os metadados de recursos ou as métricas de recursos.

### Note

Adicionar criptografia a um tópico ou fila FIFO existente não criptografa nenhuma mensagem armazenada no backlog, e remover criptografia de um tópico ou fila deixa as mensagens armazenadas em backlog criptografadas.

Os tópicos FIFO do SNS descriptografam as mensagens imediatamente antes de entregá-las aos endpoints inscritos. As filas FIFO do SQS descriptografam a mensagem antes de retorná-las à aplicação consumidora. Para obter mais informações, consulte [Criptografia de dados](#) e a publicação [Encrypting messages published to Amazon SNS with AWS KMS](#) (“Criptografar mensagens publicadas no Amazon SNS com o AWS KMS”) no Blog de computação da AWS.

Além disso, os tópicos FIFO do SNS e as filas FIFO do SQS oferecem suporte à privacidade de mensagens com [endpoints da VPC de interface](#) com tecnologia do AWS PrivateLink. Usando endpoints de interface, você pode enviar mensagens de sub-redes da Amazon Virtual Private Cloud (Amazon VPC) para tópicos e filas FIFO sem atravessar a Internet pública. Este modelo mantém suas mensagens dentro da infraestrutura e rede da AWS, o que aumenta a segurança geral da sua aplicação. Quando você usa o AWS PrivateLink, não é necessário configurar um gateway da Internet, uma conversão de endereço de rede (NAT) ou uma rede privada virtual (VPN). Para obter mais informações, consulte [Privacidade do tráfego entre redes](#) e a publicação [Securing messages published to Amazon SNS with AWS PrivateLink](#) (“Proteger mensagens publicadas no Amazon SNS com o AWS PrivateLink”) no Blog de segurança da AWS.

Os tópicos FIFO do SNS também oferecem suporte a filas de mensagens mortas e ao armazenamento de mensagens nas zonas de disponibilidade. Para obter mais informações, consulte [Durabilidade de mensagens para tópicos FIFO](#).

## Durabilidade de mensagens para tópicos FIFO

Os tópicos FIFO do Amazon SNS e as filas FIFO do Amazon SQS são duráveis. Os dois tipos de recursos armazenam mensagens de maneira redundante em várias zonas de disponibilidade e fornecem filas de mensagens mortas para lidar com casos excepcionais.

No Amazon SNS, a entrega de mensagens falha quando o tópico do Amazon SNS não pode acessar uma fila do Amazon SQS inscrita devido a um erro do lado do cliente ou um erro do lado do servidor:

- Os erros do lado do cliente ocorrem quando o tópico FIFO do Amazon SNS tem metadados de inscrição obsoletos. Duas causas comuns de erros do lado do cliente ocorrem quando o proprietário da fila do Amazon SQS executa um dos seguintes procedimentos:
  - Exclui a fila.
  - Altera a política de filas impedindo que o principal de serviço do Amazon SNS entregue mensagens a ela.

O Amazon SNS não tenta entregar mensagens que falharam devido a erros do lado do cliente.

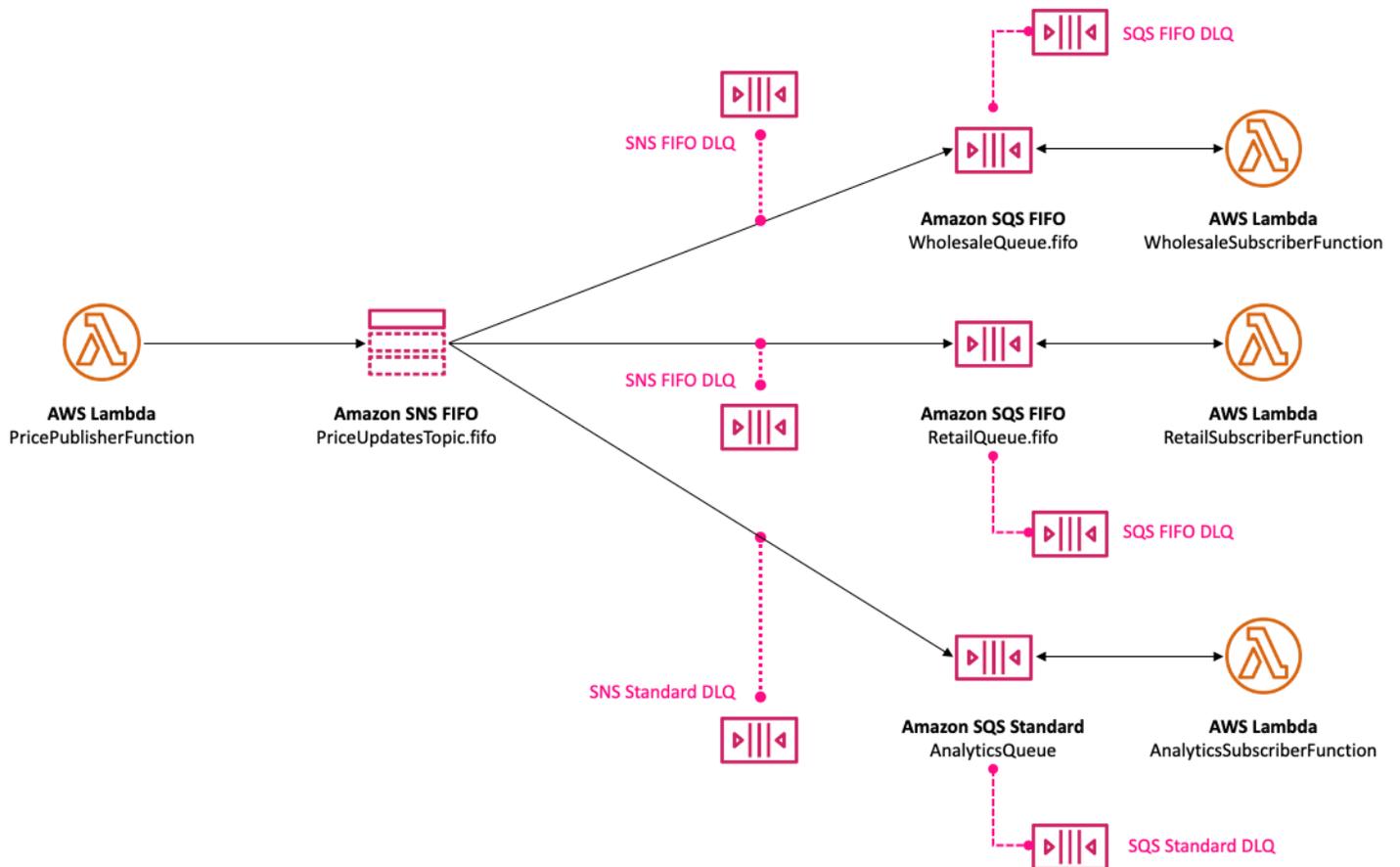
- Erros do lado do servidor podem ocorrer nestas situações:
  - O serviço Amazon SQS não está disponível.
  - O Amazon SQS falha ao processar uma solicitação válida do serviço Amazon SNS.

Quando ocorrem erros do lado do servidor, os tópicos FIFO do Amazon SNS tentam executar as entregas com falha novamente até 100.015 vezes ao longo de 23 dias. Para obter mais informações, consulte [Novas tentativas de entrega de mensagens do Amazon SNS](#).

Para qualquer tipo de erro, o Amazon SNS pode segregar mensagens nas filas de mensagens mortas do Amazon SQS para que os dados não sejam perdidos.

No Amazon SQS, o processamento de mensagens falha quando a aplicação consumidora não consegue receber a mensagem, processá-la e excluí-la da fila. Quando o número máximo de solicitações de recebimento falhar, o Amazon SQS pode colocar mensagens em filas de mensagens mortas para que os dados não sejam perdidos.

No [exemplo de caso de uso de gerenciamento de preços de peças automotivas](#), a empresa pode atribuir uma fila de mensagens não entregues (DLQ) do Amazon SQS a cada assinatura de tópico FIFO do Amazon SNS, bem como a cada fila do Amazon SQS inscrita. Isso protege a empresa contra qualquer perda de atualização de preço.



A fila de mensagens não entregues associada a uma assinatura do Amazon SNS deve ser uma fila do Amazon SQS do mesmo tipo da fila de assinatura. Por exemplo, a assinatura FIFO do Amazon SNS de uma fila FIFO do Amazon SQS deve ter uma fila FIFO do Amazon SQS como a fila de mensagens não entregues. Da mesma forma, a assinatura FIFO do Amazon SNS de uma fila padrão do Amazon SQS deve ter uma fila padrão do Amazon SQS como fila de mensagens não entregues. Para obter mais informações, consulte [Filas de mensagens mortas \(DLQs\) do Amazon SNS](#) e a publicação [Designing durable serverless apps with DLQs for Amazon SNS, Amazon SQS, AWS Lambda](#) (“Projetar aplicações sem servidor duráveis com DLQs para Amazon SNS, Amazon SQS, AWS Lambda no Blog de computação da AWS).

Para obter maior durabilidade para auxiliar na recuperação de falhas subsequentes, os proprietários de tópicos também podem usar tópicos FIFO para arquivar mensagens por até 365 dias. Os assinantes de tópicos podem reproduzir essas mensagens arquivadas em um endpoint inscrito com o objetivo de recuperar mensagens perdidas em decorrência de uma falha em uma aplicação subsequente ou para replicar o estado de uma aplicação existente. Para saber mais, consulte [Arquivamento e reprodução de mensagens de tópicos FIFO](#).

# Arquivamento e reprodução de mensagens de tópicos FIFO

## Tópicos

- [O que é arquivamento e reprodução de mensagens?](#)
- [Arquivamento de mensagens para proprietários de tópicos FIFO](#)
- [Reprodução de mensagens para assinantes de tópicos FIFO](#)

## O que é arquivamento e reprodução de mensagens?

O arquivamento e reprodução de mensagens do Amazon SNS é um arquivamento de mensagens local, sem código, que permite aos proprietários de tópicos armazenar (ou arquivar) mensagens no respectivo tópico. Os assinantes de tópicos podem então recuperar (ou reproduzir) as mensagens arquivadas de volta em um endpoint inscrito, que pode ser usado para:

- Recuperar mensagens que podem ter sido perdidas devido a uma falha em uma aplicação subsequente.
- Replicar o estado de uma aplicação existente em uma nova aplicação assinando o novo endpoint e selecionando o carimbo de data e hora desejado do qual replicar.

É possível usar o arquivamento e reprodução de mensagens com a AWS API, o SDK, o AWS CloudFormation e o AWS Management Console.

### Note

O arquivamento e reprodução de mensagens do Amazon SNS só está disponível para tópicos FIFO de aplicação para aplicação (A2A).

O arquivamento e reprodução de mensagens consiste em dois componentes principais:

1. Arquivamento de mensagens: o proprietário do tópico ativa o recurso arquivamento e reprodução em um tópico e define um período de retenção de mensagens (até 365 dias). O proprietário do tópico também pode monitorar mensagens arquivadas usando métricas do Amazon CloudWatch. Para saber mais, consulte [Arquivamento de mensagens para proprietários de tópicos FIFO](#).

2. Reprodução de mensagens: o assinante do tópico inicia a reprodução de um conjunto de mensagens do tópico para o endpoint inscrito. Para saber mais, consulte [Reprodução de mensagens para assinantes de tópicos FIFO](#).

## Arquivamento de mensagens para proprietários de tópicos FIFO

O arquivamento de mensagens oferece a capacidade de arquivar uma única cópia de todas as mensagens publicadas no tópico. É possível armazenar mensagens publicadas no tópico habilitando a política de arquivamento de mensagens no tópico, o que permite o arquivamento de mensagens para todas as assinaturas vinculadas a esse tópico. As mensagens podem ser arquivadas por um período mínimo de 1 dia a, no máximo, 365 dias.

Há cobranças adicionais ao definir uma política de arquivamento. Para obter informações sobre preços, consulte [Definição de preço do Amazon SNS](#).

### Tópicos

- [Criar uma política de arquivamento de mensagens usando o AWS Management Console](#)
- [Criar uma política de arquivamento de mensagens usando a API](#)
- [Criar uma política de arquivamento de mensagens usando o SDK](#)
- [Criar uma política de arquivamento de mensagens com o AWS CloudFormation](#)
- [Conceder acesso a um arquivo criptografado](#)
- [Monitorar métricas de arquivo de mensagens com o Amazon CloudWatch](#)

## Criar uma política de arquivamento de mensagens usando o AWS Management Console

Use essa opção para criar uma política de arquivamento de mensagens com o AWS Management Console.

1. Faça login no console [do Amazon SNS](#).
2. Selecione um tópico ou crie um. Para saber mais sobre como criar tópicos, consulte [Criar um tópico do Amazon SNS](#).

 Note

O arquivamento e reprodução de mensagens do Amazon SNS só está disponível para tópicos FIFO de aplicação para aplicação (A2A).

3. Na página Editar tópico, expanda a seção Política de arquivamento.
4. Ative o recurso Política de arquivamento e insira o número de dias durante os quais você deseja arquivar mensagens no tópico.
5. Escolha Save changes (Salvar alterações).

### Como visualizar, editar e desativar uma política de tópico de arquivamento de mensagens

- Na página Detalhes do tópico, a Política de retenção exibe o status da política de arquivamento, incluindo o número de dias para os quais ela está definida. Selecione a guia Política de arquivamento para ver os seguintes detalhes do arquivamento de mensagens:
  - Status: o status de arquivamento e reprodução aparece como ativo quando uma política de arquivamento é aplicada. O status de arquivamento e reprodução aparece como inativo quando a política de arquivamento é definida como um objeto JSON vazio.
  - Período de retenção de mensagens: o número especificado de dias de retenção de mensagens.
  - Data de início do arquivamento: a data a partir da qual os assinantes podem reproduzir as mensagens.
  - Pré-visualização de JSON: a pré-visualização JSON da política de arquivamento.
- (Opcional) Para editar uma política de arquivamento, acesse a página de resumo do tópico e selecione Editar.
- (Opcional) Para desativar uma política de arquivamento, acesse a página de resumo do tópico e selecione Editar. Desative a Política de arquivamento e selecione Salvar alterações.
- (Opcional) Para excluir um tópico com uma política de arquivamento, é necessário primeiro desativar a política de arquivamento conforme descrito anteriormente.

 Important

Para evitar exclusões acidentais de mensagens, você não pode excluir um tópico com uma política de arquivamento de mensagens ativa. A política de arquivamento de mensagens do tópico deve ser desativada para que o tópico possa ser excluído. Quando você desativa uma política de arquivamento de mensagens, o Amazon SNS exclui todas as mensagens

arquivadas. Ao excluir um tópico, as assinaturas são removidas e nenhuma mensagem em trânsito pode ser entregue.

## Criar uma política de arquivamento de mensagens usando a API

Para criar uma política de arquivamento de mensagens com a API, é necessário adicionar o atributo `ArchivePolicy` ao tópico. É possível definir uma `ArchivePolicy` usando as ações da API `CreateTopic` e `SetTopicAttributes`. A `ArchivePolicy` tem um valor único, `MessageRetentionPeriod`, que representa o número de dias pelos quais o Amazon SNS retém as mensagens. Para ativar o arquivamento de mensagens para o tópico, defina o `MessageRetentionPeriod` como um valor inteiro maior que zero. Por exemplo, para reter mensagens em seu arquivo por 30 dias, defina a `ArchivePolicy` como:

```
{
  "ArchivePolicy": {
    "MessageRetentionPeriod": "30"
  }
}
```

Para desativar o arquivamento de mensagens do tópico e limpar o arquivo, cancele a definição da `ArchivePolicy` da seguinte forma:

```
{}
```

## Criar uma política de arquivamento de mensagens usando o SDK

Para usar um AWS SDK, configure-o com suas credenciais. Para obter mais informações, consulte [Arquivos config e credentials compartilhados](#) no Guia de referência de AWS SDKs e ferramentas.

O exemplo de código a seguir mostra como definir a `ArchivePolicy` para um tópico do Amazon SNS com o objetivo de reter todas as mensagens publicadas no tópico por 30 dias.

```
// Specify the ARN of the Amazon SNS topic to set the ArchivePolicy for.
String topicArn =
    "arn:aws:sns:us-east-2:123456789012:MyArchiveTopic.fifo";
```

```
// Set the MessageRetentionPeriod to 30 days for the ArchivePolicy.
String archivePolicy =
    "{\"MessageRetentionPeriod\": \"30\"}";

// Set the ArchivePolicy for the Amazon SNS topic
SetTopicAttributesRequest request = new SetTopicAttributesRequest()
    .withTopicArn(topicArn)
    .withAttributeName("ArchivePolicy")
    .withAttributeValue(archivePolicy);
sns.setTopicAttributes(request);
```

## Criar uma política de arquivamento de mensagens com o AWS CloudFormation

Para criar uma política de arquivamento usando o AWS CloudFormation, consulte [AWS::SNS::Topic](#) no Guia do usuário do AWS CloudFormation.

## Conceder acesso a um arquivo criptografado

Para que um assinante possa começar a reproduzir mensagens de um tópico criptografado, siga as etapas abaixo. Como as mensagens passadas são reproduzidas, o Amazon SNS precisa ter acesso Decrypt provisionado à chave do KMS que foi usada para criptografar as mensagens no arquivo.

1. Ao criptografar mensagens com uma chave do KMS e armazená-las no tópico, é necessário conceder ao Amazon SNS a capacidade de descriptografar essas mensagens por meio da política de chave. Para saber mais, consulte [Conceder permissões de descriptografia ao Amazon SNS](#).
2. Habilite o AWS KMS para o Amazon SNS. Para saber mais, consulte [Configurar permissões do AWS KMS](#).

### Important

Quando você adicionar as novas seções à política de chave do KMS, não altere as seções existentes na política. Se a criptografia estiver habilitada em um tópico e a chave do KMS estiver desabilitada ou tiver sido excluída, ou a política de chave do KMS não estiver configurada corretamente para o Amazon SNS, o Amazon SNS não poderá reproduzir mensagens para seus assinantes.

## Conceder permissões decriptografia ao Amazon SNS

Para que o Amazon SNS acesse mensagens criptografadas de dentro do arquivo do tópico e as reproduza nos endpoints inscritos, é necessário habilitar o princípio de serviço do Amazon SNS para descriptografar essas mensagens.

Veja a seguir um exemplo da política que é necessária para permitir que a entidade principal do serviço do Amazon SNS descriptografe mensagens armazenadas durante uma reprodução de mensagens históricas do tópico.

```
{
  "Sid": "Allow SNS to decrypt archived messages",
  "Effect": "Allow",
  "Principal": {
    "Service": "sns.amazonaws.com"
  },
  "Action": [
    "kms:Decrypt",
    "kms:GenerateDataKey"
  ],
  "Resource": "*"
}
```

## Monitorar métricas de arquivo de mensagens com o Amazon CloudWatch

É possível monitorar mensagens arquivadas com o Amazon CloudWatch usando as métricas a seguir. Para receber notificação sobre anomalias nas workloads e ajudar a evitar impactos, é possível configurar os alarmes do Amazon CloudWatch com base nessas métricas. Para obter mais detalhes, consulte [Registrar em log e monitorar no Amazon SNS](#).

Métrica	Descrição
ApproximateNumberOfMessagesArchived	Fornece ao proprietário do tópico o número agregado de mensagens arquivadas no arquivo de tópicos, com resolução de 60 minutos.
ApproximateNumberOfBytesArchived	Fornece ao proprietário do tópico o número agregado de bytes arquivados, em todas as mensagens no arquivo de tópicos, com resolução de 60 minutos.

Métrica	Descrição
NumberOfMessagesArchiveProcessing	Fornece ao proprietário do tópico o número de mensagens salvas no arquivo do tópico durante o intervalo em resolução de 1 minuto.
NumberOfBytesArchiveProcessing	Fornece ao proprietário do tópico o número agregado de bytes salvos no arquivo do tópico durante o intervalo em resolução de 1 minuto.

A API `GetTopicAttributes` tem uma propriedade `BeginningArchiveTime`, que representa o carimbo de data e hora mais antigo no qual um assinante pode iniciar uma reprodução. O seguinte exemplo representa uma resposta para essa ação de API:

```
{
  "ArchivePolicy": {
    "MessageRetentionPeriod": "<integer>"
  },
  "BeginningArchiveTime": "<timestamp>",
  ...
}
```

## Reprodução de mensagens para assinantes de tópicos FIFO

A reprodução do Amazon SNS permite que os assinantes do tópico recuperem mensagens arquivadas do datastore do tópico e as entreguem novamente (ou as reproduzam) em um endpoint inscrito. As mensagens podem ser reproduzidas assim que a assinatura é criada. Uma mensagem reproduzida tem o mesmo conteúdo, `MessageId`, e `Timestamp` como a cópia original, e também contém o atributo `Replayed`, para ajudar você a identificar que é uma mensagem reproduzida. Para reproduzir apenas mensagens selecionadas, é possível adicionar uma política de filtro à assinatura. Para obter mais informações sobre como filtrar mensagens, consulte [Filtrar mensagens reproduzidas](#).

### Tópicos

- [Criar uma política de reprodução de mensagens com o AWS Management Console](#)
- [Adicionar uma política de reprodução à assinatura com a API](#)
- [Adicionar uma política de reprodução à assinatura com o SDK](#)

- [Filtrar mensagens reproduzidas](#)
- [Monitorar métricas de reprodução de mensagens com o Amazon CloudWatch](#)

## Criar uma política de reprodução de mensagens com o AWS Management Console

Use essa opção para criar uma política de reprodução de mensagens com o AWS Management Console.

1. Faça login no console [do Amazon SNS](#).
2. Selecione uma assinatura de tópico ou crie uma. Para saber mais sobre como criar assinaturas, consulte [Assinar tópico do Amazon SNS](#).
3. Para iniciar a reprodução de mensagens, acesse o menu suspenso Reproduzir e selecione Iniciar reprodução.
4. No modal Prazo de repetição, selecione:
  - a. Escolher data e a hora de início da reprodução: selecione a data (formato AAAA/MM/DD) e a hora (formato hh:mm:ss de 24 horas) em que você deseja começar a reproduzir as mensagens arquivadas. A hora de início deve ser posterior ao início da hora de arquivamento aproximada.
  - b. (Opcional) Escolher data e a hora de início da reprodução: selecione a data (formato AAAA/MM/DD) e a hora (formato hh:mm:ss de 24 horas) em que você deseja parar de reproduzir as mensagens arquivadas.
  - c. Selecione Iniciar repetição.
5. (Opcional) Para interromper a reprodução de mensagens, acesse a página Detalhes da assinatura e selecione Interromper repetição no menu suspenso Reproduzir.
6. (Opcional) Para monitorar as métricas de reprodução de mensagens nesse fluxo de trabalho usando o CloudWatch, consulte [Monitorar métricas de reprodução de mensagens com o Amazon CloudWatch](#).

Como visualizar e editar uma política de reprodução de mensagens

É possível realizar as seguintes ações na página Detalhes da assinatura:

- Para visualizar o status de reprodução de mensagens, o campo Status de reprodução exibe os seguintes valores:

- **Concluído:** a reprodução reenviou com êxito todas as mensagens e agora está enviando mensagens recém-publicadas.
- **Em andamento:** no momento, as mensagens selecionadas estão sendo reproduzidas.
- **Com falha:** não foi possível concluir a reprodução.
- **Pendente:** o estado padrão durante o início da reprodução.
- (Opcional) Para modificar a política de reprodução de mensagens, acesse a página [Detalhes da assinatura](#) e selecione **Iniciar reprodução** no menu suspenso **Reproduzir**. Ao iniciar uma reprodução, a reprodução existente será substituída.

## Adicionar uma política de reprodução à assinatura com a API

Para reproduzir mensagens arquivadas, use o atributo `ReplayPolicy`. A `ReplayPolicy` pode ser usada com as ações `Subscribe` e `SetSubscriptionAttributes` da API. Essa política tem os seguintes valores:

- `StartingPoint` (obrigatório): sinaliza de onde começar a reproduzir as mensagens.
- `EndingPoint` (opcional): sinaliza quando parar de reproduzir mensagens. Se `EndingPoint` for omitido, a reprodução continuará até a hora atual.
- `PointType` (obrigatório): define o tipo dos pontos de início e término. No momento, o valor compatível para `PointType` é `Timestamp`.

Por exemplo, para se recuperar de uma falha subsequente e reenviar todas as mensagens por um período de duas horas em 1.º de outubro de 2023, use a ação `SetSubscriptionAttributes` da API para definir uma `ReplayPolicy` da seguinte maneira:

```
{
  "PointType": "Timestamp",
  "StartingPoint": "2023-10-01T10:00:00.000Z",
  "EndingPoint": "2023-10-01T12:00:00.000Z"
}
```

Para reproduzir todas as mensagens enviadas ao tópico a partir de 1.º de outubro de 2023 e continuar recebendo todas as mensagens recém-publicadas no tópico, use a ação `SetSubscriptionAttributes` da API para definir uma `ReplayPolicy` na assinatura da seguinte maneira:

```
{
  "PointType": "Timestamp",
  "StartingPoint": "2023-10-01T00:00:00.000Z"
}
```

Para verificar se uma mensagem foi reproduzida, o atributo booleano `Replayed` é adicionado a cada mensagem reproduzida.

## Adicionar uma política de reprodução à assinatura com o SDK

Para usar um AWS SDK, configure-o com suas credenciais. Para obter mais informações, consulte [Arquivos config e credentials compartilhados](#) no Guia de referência de AWS SDKs e ferramentas.

O exemplo de código a seguir mostra como definir a `ReplayPolicy` em uma assinatura para reenviar mensagens do arquivo do tópico FIFO do Amazon SNS por um período de duas horas em 1.º de outubro de 2023.

```
// Specify the ARN of the Amazon SNS subscription to initiate the ReplayPolicy on.
String subscriptionArn =
    "arn:aws:sns:us-
    east-2:123456789012:MyArchiveTopic.fifo:1d2a3e9d-7f2f-447c-88ae-03f1c68294da";

// Set the ReplayPolicy to replay messages from the topic's archive
// for a 2 hour time period on October 1st 2023 between 10am and 12pm UTC.
String replayPolicy =
    "{\"PointType\": \"Timestamp\", \"StartingPoint\": \"2023-10-01T10:00:00.000Z\",
    \"EndingPoint\": \"2023-10-01T12:00:00.000Z\"}";

// Set the ArchivePolicy for the Amazon SNS topic
SetSubscriptionAttributesRequest request = new SetSubscriptionAttributesRequest()
    .withSubscriptionArn(subscriptionArn)
    .withAttributeName("ReplayPolicy")
    .withAttributeValue(replayPolicy);
sns.setSubscriptionAttributes(request);
```

## Filtrar mensagens reproduzidas

A filtragem de mensagens do Amazon SNS permite que você controle as mensagens reproduzidas que o Amazon SNS reproduz no endpoint do assinante. Quando a filtragem e o arquivamento de mensagens estão habilitados, o Amazon SNS primeiro recupera a mensagem do datastore do tópico

e, depois, aplica a mensagem à `FilterPolicy` da assinatura. A mensagem é enviada ao endpoint inscrito quando há uma correspondência. Caso contrário, ela é filtrada. Para obter mais informações, consulte [Políticas de filtro de assinatura do Amazon SNS](#).

## Monitorar métricas de reprodução de mensagens com o Amazon CloudWatch

É possível monitorar mensagens reproduzidas com o Amazon CloudWatch usando as métricas a seguir. Para receber notificação sobre anomalias nas workloads e ajudar a evitar impactos, é possível configurar os alarmes do Amazon CloudWatch com base nessas métricas. Para obter mais detalhes, consulte [Registrar em log e monitorar no Amazon SNS](#).

Métrica	Descrição
<code>NumberOfReplayedNotificationsDelivered</code>	Fornece ao assinante o número agregado de mensagens reproduzidas do arquivo de tópicos, com resolução de 1 minuto.
<code>NumberOfReplayedNotificationsFailed</code>	Fornece ao assinante o número agregado de mensagens reproduzidas que não foram entregues do arquivo de tópicos, com resolução de 1 minuto.

## Exemplos de código para tópicos FIFO

É possível usar os exemplos de código a seguir para integrar o [exemplo de caso de uso de gerenciamento de preços de peças automotivas](#) usando um tópico FIFO do Amazon SNS com uma fila padrão ou uma fila FIFO do Amazon SQS.

### Tópicos

- [Uso de um SDK da AWS](#)
- [Usar o AWS CloudFormation](#)

## Uso de um SDK da AWS

Usando um AWS SDK, você cria um tópico FIFO do Amazon SNS definindo seu atributo `FifoTopic` como **true**. Você cria uma fila FIFO do Amazon SQS definindo seu atributo `FifoQueue` como

**true**. Além disso, você deve adicionar o sufixo **.fifo** ao nome de cada recurso FIFO. Depois de criar um tópico ou fila FIFO, não é possível convertê-lo em um tópico ou fila padrão.

O seguinte exemplo de código cria esses recursos de fila FIFO e padrão:

- O tópico FIFO do Amazon SNS que distribui as atualizações de preços
- As filas FIFO do Amazon SQS que fornecem essas atualizações para as aplicações de atacado e varejo
- A fila padrão do Amazon SQS para a aplicação de análise que armazena registros, que podem ser consultados para fins de business intelligence (BI)
- As assinaturas FIFO do Amazon SNS que conectam as três filas ao tópico

Esse exemplo define [políticas de filtro](#) nas assinaturas. Se você testar o exemplo publicando uma mensagem no tópico, faça isso com o atributo `business`. Especifique `retail` ou `wholesale` como valor do atributo. Caso contrário, a mensagem será filtrada e não será entregue às filas inscritas. Para obter mais informações, consulte [Filtragem de mensagens para tópicos FIFO](#).

Java

SDK para Java 2.x

 Note

Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

Esse exemplo

- cria um tópico FIFO do Amazon SNS, duas filas FIFO do Amazon SQS e uma fila padrão.
- inscreve as filas no tópico e publica a mensagem no tópico.

O [teste](#) verifica o recebimento da mensagem em cada fila. O [exemplo completo](#) também mostra a adição de políticas de acesso e exclui os recursos no final.

```
public class PriceUpdateExample {
    public final static SnsClient snsClient = SnsClient.create();
    public final static SqsClient sqsClient = SqsClient.create();

    public static void main(String[] args) {
```

```
    final String usage = "\n" +
        "Usage: " +
        "    <topicName> <wholesaleQueueFifoName> <retailQueueFifoName>
<analyticsQueueName>\n\n" +
        "Where:\n" +
        "    fifoTopicName - The name of the FIFO topic that you want to
create. \n\n" +
        "    wholesaleQueueARN - The name of a SQS FIFO queue that will be
created for the wholesale consumer. \n\n"
        +
        "    retailQueueARN - The name of a SQS FIFO queue that will
created for the retail consumer. \n\n" +
        "    analyticsQueueARN - The name of a SQS standard queue that
will be created for the analytics consumer. \n\n";
    if (args.length != 4) {
        System.out.println(usage);
        System.exit(1);
    }

    final String fifoTopicName = args[0];
    final String wholeSaleQueueName = args[1];
    final String retailQueueName = args[2];
    final String analyticsQueueName = args[3];

    // For convenience, the QueueData class holds metadata about a queue:
    // ARN, URL,
    // name and type.
    List<QueueData> queues = List.of(
        new QueueData(wholeSaleQueueName, QueueType.FIFO),
        new QueueData(retailQueueName, QueueType.FIFO),
        new QueueData(analyticsQueueName, QueueType.Standard));

    // Create queues.
    createQueues(queues);

    // Create a topic.
    String topicARN = createFIFOTopic(fifoTopicName);

    // Subscribe each queue to the topic.
    subscribeQueues(queues, topicARN);

    // Allow the newly created topic to send messages to the queues.
    addAccessPolicyToQueuesFINAL(queues, topicARN);
```

```
// Publish a sample price update message with payload.
publishPriceUpdate(topicARN, "{\"product\": 214, \"price\": 79.99}",
"Consumables");

// Clean up resources.
deleteSubscriptions(queues);
deleteQueues(queues);
deleteTopic(topicARN);
}

public static String createFIFOTopic(String topicName) {
    try {
        // Create a FIFO topic by using the SNS service client.
        Map<String, String> topicAttributes = Map.of(
            "FifoTopic", "true",
            "ContentBasedDeduplication", "false");

        CreateTopicRequest topicRequest = CreateTopicRequest.builder()
            .name(topicName)
            .attributes(topicAttributes)
            .build();

        CreateTopicResponse response = snsClient.createTopic(topicRequest);
        String topicArn = response.topicArn();
        System.out.println("The topic ARN is" + topicArn);

        return topicArn;
    } catch (SnsException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
    return "";
}

public static void subscribeQueues(List<QueueData> queues, String topicARN) {
    queues.forEach(queue -> {
        SubscribeRequest subscribeRequest = SubscribeRequest.builder()
            .topicArn(topicARN)
            .endpoint(queue.queueARN)
            .protocol("sqs")
            .build();
    });
}
```

```
        // Subscribe to the endpoint by using the SNS service client.
        // Only Amazon SQS queues can receive notifications from an Amazon
SNS FIFO
        // topic.
        SubscribeResponse subscribeResponse =
snsClient.subscribe(subscribeRequest);
        System.out.println("The queue [" + queue.queueARN + "] subscribed to
the topic [" + topicARN + "]");
        queue.subscriptionARN = subscribeResponse.subscriptionArn();
    });
}

    public static void publishPriceUpdate(String topicArn, String payload, String
groupId) {

        try {
            // Create and publish a message that updates the wholesale price.
            String subject = "Price Update";
            String dedupId = UUID.randomUUID().toString();
            String attributeName = "business";
            String attributeValue = "wholesale";

            MessageAttributeValue msgAttValue = MessageAttributeValue.builder()
                .dataType("String")
                .stringValue(attributeValue)
                .build();

            Map<String, MessageAttributeValue> attributes = new HashMap<>();
            attributes.put(attributeName, msgAttValue);
            PublishRequest pubRequest = PublishRequest.builder()
                .topicArn(topicArn)
                .subject(subject)
                .message(payload)
                .messageGroupId(groupId)
                .messageDeduplicationId(dedupId)
                .messageAttributes(attributes)
                .build();

            final PublishResponse response = snsClient.publish(pubRequest);
            System.out.println(response.messageId());
            System.out.println(response.sequenceNumber());
            System.out.println("Message was published to " + topicArn);

        } catch (SnsException e) {
```

```
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}
```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API do AWS SDK for Java 2.x.
  - [CreateTopic](#)
  - [Publicar](#)
  - [Assinar](#)

## Python

### SDK para Python (Boto3)

#### Note

Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

Crie um tópico FIFO do Amazon SNS, inscreva filas padrão e FIFO do Amazon SQS no tópico e publique uma mensagem no tópico.

```
def usage_demo():
    """Shows how to subscribe queues to a FIFO topic."""
    print("-" * 88)
    print("Welcome to the `Subscribe queues to a FIFO topic` demo!")
    print("-" * 88)

    sns = boto3.resource("sns")
    sqs = boto3.resource("sqs")
    fifo_topic_wrapper = FifoTopicWrapper(sns)
    sns_wrapper = SnsWrapper(sns)

    prefix = "sqs-subscribe-demo-"
    queues = set()
    subscriptions = set()
```

```
wholesale_queue = sqs.create_queue(
    QueueName=prefix + "wholesale.fifo",
    Attributes={
        "MaximumMessageSize": str(4096),
        "ReceiveMessageWaitTimeSeconds": str(10),
        "VisibilityTimeout": str(300),
        "FifoQueue": str(True),
        "ContentBasedDeduplication": str(True),
    },
)
queues.add(wholesale_queue)
print(f"Created FIFO queue with URL: {wholesale_queue.url}.")

retail_queue = sqs.create_queue(
    QueueName=prefix + "retail.fifo",
    Attributes={
        "MaximumMessageSize": str(4096),
        "ReceiveMessageWaitTimeSeconds": str(10),
        "VisibilityTimeout": str(300),
        "FifoQueue": str(True),
        "ContentBasedDeduplication": str(True),
    },
)
queues.add(retail_queue)
print(f"Created FIFO queue with URL: {retail_queue.url}.")

analytics_queue = sqs.create_queue(QueueName=prefix + "analytics",
Attributes={})
queues.add(analytics_queue)
print(f"Created standard queue with URL: {analytics_queue.url}.")

topic = fifo_topic_wrapper.create_fifo_topic("price-updates-topic.fifo")
print(f"Created FIFO topic: {topic.attributes['TopicArn']}.")

for q in queues:
    fifo_topic_wrapper.add_access_policy(q, topic.attributes["TopicArn"])

print(f"Added access policies for topic: {topic.attributes['TopicArn']}.")

for q in queues:
    sub = fifo_topic_wrapper.subscribe_queue_to_topic(
        topic, q.attributes["QueueArn"]
    )
    subscriptions.add(sub)
```

```
print(f"Subscribed queues to topic: {topic.attributes['TopicArn']}.")

input("Press Enter to publish a message to the topic.")

message_id = fifo_topic_wrapper.publish_price_update(
    topic, '{"product": 214, "price": 79.99}', "Consumables"
)

print(f"Published price update with message ID: {message_id}.")

# Clean up the subscriptions, queues, and topic.
input("Press Enter to clean up resources.")
for s in subscriptions:
    sns_wrapper.delete_subscription(s)

sns_wrapper.delete_topic(topic)

for q in queues:
    fifo_topic_wrapper.delete_queue(q)

print(f"Deleted subscriptions, queues, and topic.")

print("Thanks for watching!")
print("-" * 88)

class FifoTopicWrapper:
    """Encapsulates Amazon SNS FIFO topic and subscription functions."""

    def __init__(self, sns_resource):
        """
        :param sns_resource: A Boto3 Amazon SNS resource.
        """
        self.sns_resource = sns_resource

    def create_fifo_topic(self, topic_name):
        """
        Create a FIFO topic.
        Topic names must be made up of only uppercase and lowercase ASCII
        letters,
        numbers, underscores, and hyphens, and must be between 1 and 256
        characters long.
        """
```

For a FIFO topic, the name must end with the `.fifo` suffix.

```
:param topic_name: The name for the topic.
:return: The new topic.
"""
try:
    topic = self.sns_resource.create_topic(
        Name=topic_name,
        Attributes={
            "FifoTopic": str(True),
            "ContentBasedDeduplication": str(False),
        },
    )
    logger.info("Created FIFO topic with name=%s.", topic_name)
    return topic
except ClientError as error:
    logger.exception("Couldn't create topic with name=%s!", topic_name)
    raise error

@staticmethod
def add_access_policy(queue, topic_arn):
    """
    Add the necessary access policy to a queue, so
    it can receive messages from a topic.

    :param queue: The queue resource.
    :param topic_arn: The ARN of the topic.
    :return: None.
    """
    try:
        queue.set_attributes(
            Attributes={
                "Policy": json.dumps(
                    {
                        "Version": "2012-10-17",
                        "Statement": [
                            {
                                "Sid": "test-sid",
                                "Effect": "Allow",
                                "Principal": {"AWS": "*"},
                                "Action": "SQS:SendMessage",
                                "Resource": queue.attributes["QueueArn"],
                                "Condition": {
```

```
        "ArnLike": {"aws:SourceArn": topic_arn}
    },
    ],
}
)
)
logger.info("Added trust policy to the queue.")
except ClientError as error:
    logger.exception("Couldn't add trust policy to the queue!")
    raise error

@staticmethod
def subscribe_queue_to_topic(topic, queue_arn):
    """
    Subscribe a queue to a topic.

    :param topic: The topic resource.
    :param queue_arn: The ARN of the queue.
    :return: The subscription resource.
    """
    try:
        subscription = topic.subscribe(
            Protocol="sqs",
            Endpoint=queue_arn,
        )
        logger.info("The queue is subscribed to the topic.")
        return subscription
    except ClientError as error:
        logger.exception("Couldn't subscribe queue to topic!")
        raise error

@staticmethod
def publish_price_update(topic, payload, group_id):
    """
    Compose and publish a message that updates the wholesale price.

    :param topic: The topic to publish to.
    :param payload: The message to publish.
    :param group_id: The group ID for the message.
    :return: The ID of the message.
```

```
    """
    try:
        att_dict = {"business": {"DataType": "String", "StringValue":
"wholesale"}}
        dedup_id = uuid.uuid4()
        response = topic.publish(
            Subject="Price Update",
            Message=payload,
            MessageAttributes=att_dict,
            MessageGroupId=group_id,
            MessageDeduplicationId=str(dedup_id),
        )
        message_id = response["MessageId"]
        logger.info("Published message to topic %s.", topic.arn)
    except ClientError as error:
        logger.exception("Couldn't publish message to topic %s.", topic.arn)
        raise error
    return message_id

@staticmethod
def delete_queue(queue):
    """
    Removes an SQS queue. When run against an AWS account, it can take up to
    60 seconds before the queue is actually deleted.

    :param queue: The queue to delete.
    :return: None
    """
    try:
        queue.delete()
        logger.info("Deleted queue with URL=%s.", queue.url)
    except ClientError as error:
        logger.exception("Couldn't delete queue with URL=%s!", queue.url)
        raise error
```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API do AWS SDK para Python (Boto3).
  - [CreateTopic](#)

- [Publicar](#)
- [Assinar](#)

## SAP ABAP

### SDK para SAP ABAP

#### Note

Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

Crie um tópico FIFO, inscreva uma fila FIFO do Amazon SQS no tópico e publique uma mensagem em um tópico do Amazon SNS.

```

" Creates a FIFO topic. "
DATA lt_tpc_attributes TYPE /aws1/
cl_snstopicattrsm_w=>tt_topicattributesmap.
DATA ls_tpc_attributes TYPE /aws1/
cl_snstopicattrsm_w=>ts_topicattributesmap_maprow.
ls_tpc_attributes-key = 'FifoTopic'.
ls_tpc_attributes-value = NEW /aws1/cl_snstopicattrsm_w( iv_value =
'true' ).
INSERT ls_tpc_attributes INTO TABLE lt_tpc_attributes.

TRY.
DATA(lo_create_result) = lo_sns->createtopic(
    iv_name = iv_topic_name
    it_attributes = lt_tpc_attributes
).
DATA(lv_topic_arn) = lo_create_result->get_topicarn( ).
ov_topic_arn = lv_topic_arn.
ov_topic_arn is returned for testing purposes. "
MESSAGE 'FIFO topic created' TYPE 'I'.
CATCH /aws1/cx_snstopiclimitexcdex.
MESSAGE 'Unable to create more topics. You have reached the maximum
number of topics allowed.' TYPE 'E'.
ENDTRY.

```

```

" Subscribes an endpoint to an Amazon Simple Notification Service (Amazon
SNS) topic. "
" Only Amazon Simple Queue Service (Amazon SQS) FIFO queues can be subscribed
to an SNS FIFO topic. "
TRY.
    DATA(lo_subscribe_result) = lo_sns->subscribe(
        iv_topicarn = lv_topic_arn
        iv_protocol = 'sqs'
        iv_endpoint = iv_queue_arn
    ).
    DATA(lv_subscription_arn) = lo_subscribe_result->get_subscriptionarn( ).
    ov_subscription_arn = lv_subscription_arn.
"
ov_subscription_arn is returned for testing purposes. "
    MESSAGE 'SQS queue was subscribed to SNS topic.' TYPE 'I'.
CATCH /aws1/cx_snsnotfoundexception.
    MESSAGE 'Topic does not exist.' TYPE 'E'.
CATCH /aws1/cx_snssubscriptionlmt00.
    MESSAGE 'Unable to create subscriptions. You have reached the maximum
number of subscriptions allowed.' TYPE 'E'.
ENDTRY.

" Publish message to SNS topic. "
TRY.
    DATA lt_msg_attributes TYPE /aws1/
cl_snsmessageattrvalue=>tt_messageattributemap.
    DATA ls_msg_attributes TYPE /aws1/
cl_snsmessageattrvalue=>ts_messageattributemap_maprow.
    ls_msg_attributes-key = 'Importance'.
    ls_msg_attributes-value = NEW /aws1/cl_snsmessageattrvalue( iv_datatype =
'String' iv_stringvalue = 'High' ).
    INSERT ls_msg_attributes INTO TABLE lt_msg_attributes.

    DATA(lo_result) = lo_sns->publish(
        iv_topicarn = lv_topic_arn
        iv_message = 'The price of your mobile plan has been increased from
$19 to $23'
        iv_subject = 'Changes to mobile plan'
        iv_messagegroupid = 'Update-2'
        iv_messagededuplicationid = 'Update-2.1'
        it_messageattributes = lt_msg_attributes
    ).
    ov_message_id = lo_result->get_messageid( ).
"
ov_message_id is returned for testing purposes. "
    MESSAGE 'Message was published to SNS topic.' TYPE 'I'.

```

```
CATCH /aws1/cx_snsnotfoundexception.  
    MESSAGE 'Topic does not exist.' TYPE 'E'.  
ENDTRY.
```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API do AWS SDK para SAP ABAP.
  - [CreateTopic](#)
  - [Publicar](#)
  - [Assinar](#)

## Receber mensagens de assinaturas FIFO

Agora você pode receber atualizações de preços nas três aplicações inscritas. Como mostrado no [the section called “Caso de uso de tópicos FIFO”](#), o ponto de entrada para cada aplicação consumidora é a fila do Amazon SQS, que a função do AWS Lambda correspondente pode sondar de forma automática. Quando uma fila do Amazon SQS é uma fonte de eventos para uma função do Lambda, o Lambda escala sua frota de sondas conforme necessário para consumir mensagens de forma eficiente.

Para obter mais informações, consulte [Usar o AWS Lambda com o Amazon SQS](#) no Guia do desenvolvedor do AWS Lambda. Para obter informações sobre como escrever suas próprias sondas de fila, consulte [Recommendations for Amazon SQS standard and FIFO queues](#) (“Recomendações para filas padrão e FIFO do Amazon SQS”) no Guia do desenvolvedor do Amazon Simple Queue Service e [ReceiveMessage](#) na Referência da API do Amazon Simple Queue Service.

## Usar o AWS CloudFormation

O AWS CloudFormation permite que você use um arquivo de modelo para criar e configurar um conjunto de recursos da AWS como uma unidade única. Esta seção tem um modelo de exemplo que cria o seguinte:

- O tópico FIFO do Amazon SNS que distribui as atualizações de preços
- As filas FIFO do Amazon SQS que fornecem essas atualizações para as aplicações de atacado e varejo
- A fila padrão do Amazon SQS para a aplicação de análise que armazena registros, que podem ser consultados para fins de business intelligence (BI)

- As assinaturas FIFO do Amazon SNS que conectam as três filas ao tópico
- A [política de filtros](#) que especifica que as aplicações de assinante recebem somente as atualizações de preço de que precisam

### Note

Se você testar esse exemplo de código publicando uma mensagem no tópico, faça isso com o atributo `business`. Especifique `retail` ou `wholesale` como valor do atributo. Caso contrário, a mensagem será filtrada e não será entregue às filas inscritas.

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Resources": {
    "PriceUpdatesTopic": {
      "Type": "AWS::SNS::Topic",
      "Properties": {
        "TopicName": "PriceUpdatesTopic.fifo",
        "FifoTopic": true,
        "ContentBasedDeduplication": false,
        "ArchivePolicy": {
          "MessageRetentionPeriod": "30"
        }
      }
    },
    "WholesaleQueue": {
      "Type": "AWS::SQS::Queue",
      "Properties": {
        "QueueName": "WholesaleQueue.fifo",
        "FifoQueue": true,
        "ContentBasedDeduplication": false
      }
    },
    "RetailQueue": {
      "Type": "AWS::SQS::Queue",
      "Properties": {
        "QueueName": "RetailQueue.fifo",
        "FifoQueue": true,
        "ContentBasedDeduplication": false
      }
    }
  },
}
```

```
"AnalyticsQueue": {
  "Type": "AWS::SQS::Queue",
  "Properties": {
    "QueueName": "AnalyticsQueue"
  }
},
"WholesaleSubscription": {
  "Type": "AWS::SNS::Subscription",
  "Properties": {
    "TopicArn": {
      "Ref": "PriceUpdatesTopic"
    },
    "Endpoint": {
      "Fn::GetAtt": [
        "WholesaleQueue",
        "Arn"
      ]
    },
    "Protocol": "sqs",
    "RawMessageDelivery": "false",
    "FilterPolicyScope": "MessageBody",
    "FilterPolicy": {
      "business": [
        "wholesale"
      ]
    }
  }
},
"RetailSubscription": {
  "Type": "AWS::SNS::Subscription",
  "Properties": {
    "TopicArn": {
      "Ref": "PriceUpdatesTopic"
    },
    "Endpoint": {
      "Fn::GetAtt": [
        "RetailQueue",
        "Arn"
      ]
    },
    "Protocol": "sqs",
    "RawMessageDelivery": "false",
    "FilterPolicyScope": "MessageBody",
    "FilterPolicy": {
```

```
        "business": [
            "retail"
        ]
    }
},
"AnalyticsSubscription": {
    "Type": "AWS::SNS::Subscription",
    "Properties": {
        "TopicArn": {
            "Ref": "PriceUpdatesTopic"
        },
        "Endpoint": {
            "Fn::GetAtt": [
                "AnalyticsQueue",
                "Arn"
            ]
        },
        "Protocol": "sqs",
        "RawMessageDelivery": "false"
    }
},
"SalesQueuesPolicy": {
    "Type": "AWS::SQS::QueuePolicy",
    "Properties": {
        "PolicyDocument": {
            "Statement": [
                {
                    "Effect": "Allow",
                    "Principal": {
                        "Service": "sns.amazonaws.com"
                    },
                    "Action": [
                        "sqs:SendMessage"
                    ],
                    "Resource": "*",
                    "Condition": {
                        "ArnEquals": {
                            "aws:SourceArn": {
                                "Ref": "PriceUpdatesTopic"
                            }
                        }
                    }
                }
            ]
        }
    }
}
```

```
    ]
  },
  "Queues": [
    {
      "Ref": "WholesaleQueue"
    },
    {
      "Ref": "RetailQueue"
    },
    {
      "Ref": "AnalyticsQueue"
    }
  ]
}
}
```

Para obter mais informações sobre a implantação de recursos da AWS usando um modelo do AWS CloudFormation, consulte [Get Started](#) (“Conceitos básicos”) no Manual do usuário do AWS CloudFormation.

# Publicação de mensagens do Amazon SNS

Depois de [criar um tópico do Amazon SNS](#) e [inscrever](#) um endpoint nele, é possível publicar mensagens no tópico. Quando uma mensagem é publicada, o Amazon SNS tenta entregar a mensagem aos [endpoints](#) inscritos.

## Tópicos

- [Para publicar mensagens nos tópicos do Amazon SNS usando o AWS Management Console](#)
- [Para publicar uma mensagem em um tópico usando um AWS SDK](#)
- [Publicar mensagens grandes com o Amazon SNS e o Amazon S3](#)
- [Atributos de mensagem do Amazon SNS](#)
- [Agrupamento de mensagens do Amazon SNS em lotes](#)

## Para publicar mensagens nos tópicos do Amazon SNS usando o AWS Management Console

1. Faça login no [console do Amazon SNS](#).
2. No painel de navegação à esquerda, selecione Tópicos.
3. Na página Topics (Tópicos), selecione um tópico e escolha Publish message (Publicar em um tópico).

O console abre a página Publish message to topic (Publicar mensagem no tópico).

4. Na seção Basic details (Detalhes básicos) faça o seguinte:
  - a. (Opcional) Insira o Subject (Assunto) de uma mensagem.
  - b. Para um [tópico FIFO](#), insira um Message group ID (ID do grupo de mensagens). As mensagens no mesmo grupo de mensagens são entregues na ordem em que são publicadas.
  - c. Para um tópico FIFO, insira um Message deduplication ID (ID de deduplicação de mensagem). Este ID é opcional se você habilitou a configuração de Desduplicação de mensagens baseada em conteúdo para o tópico.
  - d. (Opcional) Para [notificações por push para dispositivos móveis](#), insira um valor para Time to Live (TTL) (Vida útil (TTL)) em segundos. Essa é a quantidade de tempo que um serviço

de notificação por push, como o Apple Push Notification Service (APNs) ou Firebase Cloud Messaging (FCM), tem para entregar a mensagem ao endpoint.

5. Na seção Message body (Corpo da mensagem), siga um destes procedimentos:
  - a. Escolha Identical payload for all delivery protocols (Carga útil idêntica para todos os protocolos de entrega) e, em seguida, insira uma mensagem.
  - b. Escolha Custom payload for each delivery protocol (Carga útil personalizada para cada protocolo de entrega) e, depois, insira um objeto JSON para definir a mensagem a ser enviada a cada protocolo de entrega.

Para ter mais informações, consulte [Publicação com cargas úteis específicas da plataforma](#).

6. Na seção Message attributes (Atributos da mensagem), adicione todos os atributos que deseja que o Amazon SNS corresponda ao atributo de assinatura FilterPolicy para decidir se o endpoint inscrito está interessado na mensagem publicada.
  - a. Em Type Tipo, escolha um tipo de atributo, como String.Array.

 Note

Para o tipo de atributo String.Array, coloque a matriz entre colchetes ([]). Dentro da matriz, coloque valores de strings entre aspas duplas. As aspas não são necessárias para números nem para as palavras-chave true, false e null.

- b. Insira um atributo Name (Nome), por exemplo, customer\_interests.
  - c. Insira um atributo Value (Valor), por exemplo, ["soccer", "rugby", "hockey"].

Se o tipo do atributo for String, String.Array ou Number, o Amazon SNS avaliará o atributo de mensagem em relação à [política de filtro](#) de uma assinatura (se existir) antes de enviar a mensagem à assinatura, considerando que o escopo da política de filtro não está explicitamente definido como MessageBody.

Para ter mais informações, consulte [Atributos de mensagem do Amazon SNS](#).

7. Selecione Publish message (Publicar mensagem).

A mensagem é publicada no tópico e o console abre a página Details (Detalhes).

# Para publicar uma mensagem em um tópico usando um AWS SDK

Para usar um AWS SDK, você deve configurá-lo com suas credenciais. Para obter mais informações, consulte [The shared config and credentials files](#) (“Os arquivos compartilhados de configuração e de credenciais”) no Guia de referência de AWS SDKs e ferramentas.

Os exemplos de código a seguir mostram como usar `Publish`.

.NET

## AWS SDK for .NET

### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

Publique uma mensagem em um tópico.

```
using System;
using System.Threading.Tasks;
using Amazon.SimpleNotificationService;
using Amazon.SimpleNotificationService.Model;

/// <summary>
/// This example publishes a message to an Amazon Simple Notification
/// Service (Amazon SNS) topic.
/// </summary>
public class PublishToSNSTopic
{
    public static async Task Main()
    {
        string topicArn = "arn:aws:sns:us-
east-2:000000000000:ExampleSNSTopic";
        string messageText = "This is an example message to publish to the
ExampleSNSTopic.";

        IAmazonSimpleNotificationService client = new
AmazonSimpleNotificationServiceClient();

        await PublishToTopicAsync(client, topicArn, messageText);
    }
}
```

```
    }

    /// <summary>
    /// Publishes a message to an Amazon SNS topic.
    /// </summary>
    /// <param name="client">The initialized client object used to publish
    /// to the Amazon SNS topic.</param>
    /// <param name="topicArn">The ARN of the topic.</param>
    /// <param name="messageText">The text of the message.</param>
    public static async Task PublishToTopicAsync(
        IAmazonSimpleNotificationService client,
        string topicArn,
        string messageText)
    {
        var request = new PublishRequest
        {
            TopicArn = topicArn,
            Message = messageText,
        };

        var response = await client.PublishAsync(request);

        Console.WriteLine($"Successfully published message ID:
{response.MessageId}");
    }
}
```

Publique uma mensagem em um tópico com opções de grupo, duplicação e atributo.

```
/// <summary>
/// Publish messages using user settings.
/// </summary>
/// <returns>Async task.</returns>
public static async Task PublishMessages()
{
    Console.WriteLine("Now we can publish messages.");

    var keepSendingMessages = true;
    string? deduplicationId = null;
    string? toneAttribute = null;
    while (keepSendingMessages)
```

```
{
    Console.WriteLine();
    var message = GetUserResponse("Enter a message to publish.", "This is
a sample message");

    if (_useFifoTopic)
    {
        Console.WriteLine("Because you are using a FIFO topic, you must
set a message group ID." +
            "\r\nAll messages within the same group will be
received in the order " +
            "they were published.");

        Console.WriteLine();
        var messageGroupId = GetUserResponse("Enter a message group ID
for this message:", "1");

        if (!_useContentBasedDeduplication)
        {
            Console.WriteLine("Because you are not using content-based
deduplication, " +
                "you must enter a deduplication ID.");

            Console.WriteLine("Enter a deduplication ID for this
message.");
            deduplicationId = GetUserResponse("Enter a deduplication ID
for this message.", "1");
        }

        if (GetYesNoResponse("Add an attribute to this message?"))
        {
            Console.WriteLine("Enter a number for an attribute.");
            for (int i = 0; i < _tones.Length; i++)
            {
                Console.WriteLine($"{i + 1}. {_tones[i]}");
            }

            var selection = GetUserResponse("", "1");
            int.TryParse(selection, out var selectionNumber);

            if (selectionNumber > 0 && selectionNumber < _tones.Length)
            {
                toneAttribute = _tones[selectionNumber - 1];
            }
        }
    }
}
```

```

        }

        var messageID = await SnsWrapper.PublishToTopicWithAttribute(
            _topicArn, message, "tone", toneAttribute, deduplicationId,
messageGroupId);

        Console.WriteLine($"Message published with id {messageID}.");
    }

    keepSendingMessages = GetYesNoResponse("Send another message?",
false);
    }
}

```

Aplice as seleções do usuário à ação de publicação.

```

/// <summary>
/// Publish a message to a topic with an attribute and optional deduplication
and group IDs.
/// </summary>
/// <param name="topicArn">The ARN of the topic.</param>
/// <param name="message">The message to publish.</param>
/// <param name="attributeName">The optional attribute for the message.</
param>
/// <param name="attributeValue">The optional attribute value for the
message.</param>
/// <param name="deduplicationId">The optional deduplication ID for the
message.</param>
/// <param name="groupId">The optional group ID for the message.</param>
/// <returns>The ID of the message published.</returns>
public async Task<string> PublishToTopicWithAttribute(
    string topicArn,
    string message,
    string? attributeName = null,
    string? attributeValue = null,
    string? deduplicationId = null,
    string? groupId = null)
{
    var publishRequest = new PublishRequest()
    {
        TopicArn = topicArn,
        Message = message,

```

```

        MessageDeduplicationId = deduplicationId,
        MessageGroupId = groupId
    };

    if (attributeValue != null)
    {
        // Add the string attribute if it exists.
        publishRequest.MessageAttributes =
            new Dictionary<string, MessageAttributeValue>
            {
                { attributeName!, new MessageAttributeValue() { StringValue =
attributeValue, DataType = "String"} }
            };
    }

    var publishResponse = await
    _amazonSNSClient.PublishAsync(publishRequest);
    return publishResponse.MessageId;
}

```

- Para obter detalhes da API, consulte [Publish](#) na Referência da API do AWS SDK for .NET .

## C++

### SDK for C++

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```

/*! Send a message to an Amazon Simple Notification Service (Amazon SNS) topic.
 *!
 *! \param message: The message to publish.
 *! \param topicARN: The Amazon Resource Name (ARN) for an Amazon SNS topic.
 *! \param clientConfiguration: AWS client configuration.
 *! \return bool: Function succeeded.
 */
bool AwsDoc::SNS::publishToTopic(const Aws::String &message,
                                const Aws::String &topicARN,

```

```

                                const Aws::Client::ClientConfiguration
&clientConfiguration) {
    Aws::SNS::SNSClient snsClient(clientConfiguration);

    Aws::SNS::Model::PublishRequest request;
    request.SetMessage(message);
    request.SetTopicArn(topicARN);

    const Aws::SNS::Model::PublishOutcome outcome = snsClient.Publish(request);

    if (outcome.IsSuccess()) {
        std::cout << "Message published successfully with id '"
                    << outcome.GetResult().GetMessageId() << "'." << std::endl;
    }
    else {
        std::cerr << "Error while publishing message "
                    << outcome.GetError().GetMessage()
                    << std::endl;
    }

    return outcome.IsSuccess();
}

```

Publique uma mensagem com um atributo.

```

    static const Aws::String TONE_ATTRIBUTE("tone");
    static const Aws::Vector<Aws::String> TONES = {"cheerful", "funny",
"serious",
                                                    "sincere"};

    Aws::Client::ClientConfiguration clientConfig;
    // Optional: Set to the AWS Region (overrides config file).
    // clientConfig.region = "us-east-1";

    Aws::SNS::SNSClient snsClient(clientConfiguration);

    Aws::SNS::Model::PublishRequest request;
    request.SetTopicArn(topicARN);
    Aws::String message = askQuestion("Enter a message text to publish. ");
    request.SetMessage(message);

    if (filteringMessages && askYesNoQuestion(

```

```

        "Add an attribute to this message? (y/n) ") {
    for (size_t i = 0; i < TONES.size(); ++i) {
        std::cout << " " << (i + 1) << ". " << TONES[i] << std::endl;
    }
    int selection = askQuestionForIntRange(
        "Enter a number for an attribute. ",
        1, static_cast<int>(TONES.size()));
    Aws::SNS::Model::MessageAttributeValue messageAttributeValue;
    messageAttributeValue.SetDataType("String");
    messageAttributeValue.SetStringValue(TONES[selection - 1]);
    request.AddMessageAttributes(TONE_ATTRIBUTE, messageAttributeValue);
}

Aws::SNS::Model::PublishOutcome outcome = snsClient.Publish(request);

if (outcome.IsSuccess()) {
    std::cout << "Your message was successfully published." << std::endl;
}
else {
    std::cerr << "Error with TopicsAndQueues::Publish. "
        << outcome.GetError().GetMessage()
        << std::endl;

    cleanUp(topicARN,
        queueURLS,
        subscriptionARNS,
        snsClient,
        sqsClient);

    return false;
}

```

- Para obter detalhes da API, consulte [Publish](#) na Referência da API do AWS SDK for C++ .

## CLI

### AWS CLI

Exemplo 1: Para publicar uma mensagem em um tópico:

O exemplo `publish` a seguir publica a mensagem específica no tópico do SNS especificado. A mensagem é proveniente de um arquivo de texto, o que permite incluir quebras de linha.

```
aws sns publish \  
  --topic-arn "arn:aws:sns:us-west-2:123456789012:my-topic" \  
  --message file://message.txt
```

Conteúdo de message.txt:

```
Hello World  
Second Line
```

Saída:

```
{  
  "MessageId": "123a45b6-7890-12c3-45d6-111122223333"  
}
```

Exemplo 2: Para publicar uma mensagem SMS em um número de telefone

O exemplo publish a seguir publica a mensagem Hello world! no número de telefone +1-555-555-0100.

```
aws sns publish \  
  --message "Hello world!" \  
  --phone-number +1-555-555-0100
```

Saída:

```
{  
  "MessageId": "123a45b6-7890-12c3-45d6-333322221111"  
}
```

- Para obter detalhes da API, consulte [Publish](#) na Referência de comandos da AWS CLI .

## Go

## SDK for Go V2

 Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```
// SnsActions encapsulates the Amazon Simple Notification Service (Amazon SNS)
// actions
// used in the examples.
type SnsActions struct {
    SnsClient *sns.Client
}

// Publish publishes a message to an Amazon SNS topic. The message is then sent
// to all
// subscribers. When the topic is a FIFO topic, the message must also contain a
// group ID
// and, when ID-based deduplication is used, a deduplication ID. An optional key-
// value
// filter attribute can be specified so that the message can be filtered
// according to
// a filter policy.
func (actor SnsActions) Publish(topicArn string, message string, groupId string,
    dedupId string, filterKey string, filterValue string) error {
    publishInput := sns.PublishInput{TopicArn: aws.String(topicArn), Message:
    aws.String(message)}
    if groupId != "" {
        publishInput.MessageGroupId = aws.String(groupId)
    }
    if dedupId != "" {
        publishInput.MessageDeduplicationId = aws.String(dedupId)
    }
    if filterKey != "" && filterValue != "" {
        publishInput.MessageAttributes = map[string]types.MessageAttributeValue{
```

```
    filterKey: {DataType: aws.String("String"), StringValue:
aws.String(filterValue)},
  }
}
_, err := actor.SnsClient.Publish(context.TODO(), &publishInput)
if err != nil {
  log.Printf("Couldn't publish message to topic %v. Here's why: %v", topicArn,
err)
}
return err
}
```

- Para obter detalhes da API, consulte [Publish](#) na Referência da API do AWS SDK for Go .

## Java

### SDK para Java 2.x

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.sns.SnsClient;
import software.amazon.awssdk.services.sns.model.PublishRequest;
import software.amazon.awssdk.services.sns.model.PublishResponse;
import software.amazon.awssdk.services.sns.model.SnsException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
 * started.html
 */
public class PublishTopic {
```

```
public static void main(String[] args) {
    final String usage = ""

        Usage:    <message> <topicArn>

        Where:
            message - The message text to send.
            topicArn - The ARN of the topic to publish.
        """;

    if (args.length != 2) {
        System.out.println(usage);
        System.exit(1);
    }

    String message = args[0];
    String topicArn = args[1];
    SnsClient snsClient = SnsClient.builder()
        .region(Region.US_EAST_1)
        .build();
    pubTopic(snsClient, message, topicArn);
    snsClient.close();
}

public static void pubTopic(SnsClient snsClient, String message, String
topicArn) {
    try {
        PublishRequest request = PublishRequest.builder()
            .message(message)
            .topicArn(topicArn)
            .build();

        PublishResponse result = snsClient.publish(request);
        System.out
            .println(result.messageId() + " Message sent. Status is " +
result.sdkHttpResponse().statusCode());

    } catch (SnsException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}
}
```

- Para obter detalhes da API, consulte [Publish](#) na Referência da API do AWS SDK for Java 2.x .

## JavaScript

### SDK para JavaScript (v3)

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

Crie o cliente em um módulo separado e exporte-o.

```
import { SNSClient } from "@aws-sdk/client-sns";

// The AWS Region can be provided here using the `region` property. If you leave
// it blank
// the SDK will default to the region set in your AWS config.
export const snsClient = new SNSClient({});
```

Importe o SDK e os módulos do cliente e chame a API.

```
import { PublishCommand } from "@aws-sdk/client-sns";
import { snsClient } from "../libs/snsClient.js";

/**
 * @param {string | Record<string, any>} message - The message to send. Can be a
plain string or an object
 *
 * if you are using the `json`
`MessageStructure`.
 * @param {string} topicArn - The ARN of the topic to which you would like to
publish.
 */
export const publish = async (
  message = "Hello from SNS!",
  topicArn = "TOPIC_ARN",
```

```
) => {
  const response = await snsClient.send(
    new PublishCommand({
      Message: message,
      TopicArn: topicArn,
    }),
  );
  console.log(response);
  // {
  //   '$metadata': {
  //     httpStatusCode: 200,
  //     requestId: 'e7f77526-e295-5325-9ee4-281a43ad1f05',
  //     extendedRequestId: undefined,
  //     cfId: undefined,
  //     attempts: 1,
  //     totalRetryDelay: 0
  //   },
  //   MessageId: 'xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxxx'
  // }
  return response;
};
```

Publique uma mensagem em um tópico com opções de grupo, duplicação e atributo.

```
async publishMessages() {
  const message = await this.prompter.input({
    message: MESSAGES.publishMessagePrompt,
  });

  let groupId, deduplicationId, choices;

  if (this.isFifo) {
    await this.logger.log(MESSAGES.groupIdNotice);
    groupId = await this.prompter.input({
      message: MESSAGES.groupIdPrompt,
    });
  }

  if (this.autoDedup === false) {
    await this.logger.log(MESSAGES.deduplicationIdNotice);
    deduplicationId = await this.prompter.input({
      message: MESSAGES.deduplicationIdPrompt,
    });
  }
}
```

```
    }

    choices = await this.prompter.checkbox({
      message: MESSAGES.messageAttributesPrompt,
      choices: toneChoices,
    });
  }

  await this.snsClient.send(
    new PublishCommand({
      TopicArn: this.topicArn,
      Message: message,
      ...(groupId
        ? {
            MessageGroupId: groupId,
          }
        : {}),
      ...(deduplicationId
        ? {
            MessageDeduplicationId: deduplicationId,
          }
        : {}),
      ...(choices
        ? {
            MessageAttributes: {
              tone: {
                DataType: "String.Array",
                StringValue: JSON.stringify(choices),
              },
            },
          }
        : {}),
    })),
  );

  const publishAnother = await this.prompter.confirm({
    message: MESSAGES.publishAnother,
  });

  if (publishAnother) {
    await this.publishMessages();
  }
}
```

- Para obter mais informações, consulte o [Guia do desenvolvedor do AWS SDK for JavaScript](#).
- Para obter detalhes da API, consulte [Publish](#) na Referência da API do AWS SDK for JavaScript .

## Kotlin

### SDK for Kotlin

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```
suspend fun pubTopic(topicArnVal: String, messageVal: String) {  
  
    val request = PublishRequest {  
        message = messageVal  
        topicArn = topicArnVal  
    }  
  
    SnsClient { region = "us-east-1" }.use { snsClient ->  
        val result = snsClient.publish(request)  
        println("${result.messageId} message sent.")  
    }  
}
```

- Para obter detalhes da API, consulte [Publish](#) na Referência da API do AWS SDK para Kotlin.

## PHP

## SDK para PHP

 Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
require 'vendor/autoload.php';

use Aws\Exception\AwsException;
use Aws\Sns\SnsClient;

/**
 * Sends a message to an Amazon SNS topic.
 *
 * This code expects that you have AWS credentials set up per:
 * https://docs.aws.amazon.com/sdk-for-php/v3/developer-guide/
 * guide_credentials.html
 */

$SnSClient = new SnsClient([
    'profile' => 'default',
    'region' => 'us-east-1',
    'version' => '2010-03-31'
]);

$message = 'This message is sent from a Amazon SNS code sample.';
$topic = 'arn:aws:sns:us-east-1:111122223333:MyTopic';

try {
    $result = $SnSClient->publish([
        'Message' => $message,
        'TopicArn' => $topic,
    ]);
    var_dump($result);
} catch (AwsException $e) {
    // output error message if fails
    error_log($e->getMessage());
}
```

```
}
```

- Para obter mais informações, consulte o [Guia do desenvolvedor do AWS SDK for PHP](#).
- Para obter detalhes da API, consulte [Publish](#) na Referência da API do AWS SDK for PHP .

## PowerShell

### Ferramentas para PowerShell

Exemplo 1: Este exemplo mostra a publicação de uma mensagem com uma única `MessageAttribute` declaração em linha.

```
Publish-SNSMessage -TopicArn "arn:aws:sns:us-west-2:123456789012:my-topic" -  
Message "Hello" -MessageAttribute  
@{'City'=[Amazon.SimpleNotificationService.Model.MessageAttributeValue]@{DataType='String'  
StringValue = 'AnyCity'}}
```

Exemplo 2: Este exemplo mostra a publicação de uma mensagem com várias `MessageAttributes` declaradas com antecedência.

```
$cityAttributeValue = New-Object  
    Amazon.SimpleNotificationService.Model.MessageAttributeValue  
$cityAttributeValue.DataType = "String"  
$cityAttributeValue.StringValue = "AnyCity"  
  
$populationAttributeValue = New-Object  
    Amazon.SimpleNotificationService.Model.MessageAttributeValue  
$populationAttributeValue.DataType = "Number"  
$populationAttributeValue.StringValue = "1250800"  
  
$messageAttributes = New-Object System.Collections.Hashtable  
$messageAttributes.Add("City", $cityAttributeValue)  
$messageAttributes.Add("Population", $populationAttributeValue)  
  
Publish-SNSMessage -TopicArn "arn:aws:sns:us-west-2:123456789012:my-topic" -  
Message "Hello" -MessageAttribute $messageAttributes
```

- Para obter detalhes da API, consulte [Publicar](#) no AWS Tools for PowerShell Cmdlet Reference.

## Python

### SDK para Python (Boto3)

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

Publique uma mensagem com atributos para que uma assinatura possa filtrar com base em atributos.

```
class SnsWrapper:
    """Encapsulates Amazon SNS topic and subscription functions."""

    def __init__(self, sns_resource):
        """
        :param sns_resource: A Boto3 Amazon SNS resource.
        """
        self.sns_resource = sns_resource

    @staticmethod
    def publish_message(topic, message, attributes):
        """
        Publishes a message, with attributes, to a topic. Subscriptions can be
        filtered
        based on message attributes so that a subscription receives messages only
        when specified attributes are present.

        :param topic: The topic to publish to.
        :param message: The message to publish.
        :param attributes: The key-value attributes to attach to the message.
        Values
            must be either `str` or `bytes`.
        :return: The ID of the message.
        """
        try:
            att_dict = {}
            for key, value in attributes.items():
                if isinstance(value, str):
```

```

        att_dict[key] = {"DataType": "String", "StringValue": value}
    elif isinstance(value, bytes):
        att_dict[key] = {"DataType": "Binary", "BinaryValue": value}
    response = topic.publish(Message=message, MessageAttributes=att_dict)
    message_id = response["MessageId"]
    logger.info(
        "Published message with attributes %s to topic %s.",
        attributes,
        topic.arn,
    )
except ClientError:
    logger.exception("Couldn't publish message to topic %s.", topic.arn)
    raise
else:
    return message_id

```

Publique uma mensagem que assume diferentes formas com base no protocolo do assinante.

```

class SnsWrapper:
    """Encapsulates Amazon SNS topic and subscription functions."""

    def __init__(self, sns_resource):
        """
        :param sns_resource: A Boto3 Amazon SNS resource.
        """
        self.sns_resource = sns_resource

    @staticmethod
    def publish_multi_message(
        topic, subject, default_message, sms_message, email_message
    ):
        """
        Publishes a multi-format message to a topic. A multi-format message takes
        different forms based on the protocol of the subscriber. For example,
        an SMS subscriber might receive a short version of the message
        while an email subscriber could receive a longer version.

        :param topic: The topic to publish to.
        :param subject: The subject of the message.

```

```
is
    :param default_message: The default version of the message. This version
    sent to subscribers that have protocols that are
not
    otherwise specified in the structured message.
    :param sms_message: The version of the message sent to SMS subscribers.
    :param email_message: The version of the message sent to email
subscribers.
    :return: The ID of the message.
    """
    try:
        message = {
            "default": default_message,
            "sms": sms_message,
            "email": email_message,
        }
        response = topic.publish(
            Message=json.dumps(message), Subject=subject,
MessageStructure="json"
        )
        message_id = response["MessageId"]
        logger.info("Published multi-format message to topic %s.", topic.arn)
    except ClientError:
        logger.exception("Couldn't publish message to topic %s.", topic.arn)
        raise
    else:
        return message_id
```

- Para obter detalhes da API, consulte [Publish](#) na Referência da API do AWS SDK for Python (Boto3).

## Ruby

### SDK for Ruby

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
# Service class for sending messages using Amazon Simple Notification Service
(SNS)
class SnsMessageSender
  # Initializes the SnsMessageSender with an SNS client
  #
  # @param sns_client [Aws::SNS::Client] The SNS client
  def initialize(sns_client)
    @sns_client = sns_client
    @logger = Logger.new($stdout)
  end

  # Sends a message to a specified SNS topic
  #
  # @param topic_arn [String] The ARN of the SNS topic
  # @param message [String] The message to send
  # @return [Boolean] true if message was successfully sent, false otherwise
  def send_message(topic_arn, message)
    @sns_client.publish(topic_arn: topic_arn, message: message)
    @logger.info("Message sent successfully to #{topic_arn}.")
    true
  rescue Aws::SNS::Errors::ServiceError => e
    @logger.error("Error while sending the message: #{e.message}")
    false
  end
end

# Example usage:
if $PROGRAM_NAME == __FILE__
  topic_arn = "SNS_TOPIC_ARN" # Should be replaced with a real topic ARN
  message = "MESSAGE"        # Should be replaced with the actual message
  content
```

```
sns_client = Aws::SNS::Client.new
message_sender = SnsMessageSender.new(sns_client)

@logger.info("Sending message.")
unless message_sender.send_message(topic_arn, message)
  @logger.error("Message sending failed. Stopping program.")
  exit 1
end
end
```

- Para obter mais informações, consulte o [Guia do desenvolvedor do AWS SDK for Ruby](#).
- Para obter detalhes da API, consulte [Publish](#) na Referência da API do AWS SDK for Ruby .

## Rust

### SDK for Rust

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```
async fn subscribe_and_publish(
  client: &Client,
  topic_arn: &str,
  email_address: &str,
) -> Result<(), Error> {
  println!("Receiving on topic with ARN: `{}`", topic_arn);

  let rsp = client
    .subscribe()
    .topic_arn(topic_arn)
    .protocol("email")
    .endpoint(email_address)
    .send()
    .await?;

  println!("Added a subscription: {:?}", rsp);
```

```
let rsp = client
    .publish()
    .topic_arn(topic_arn)
    .message("hello sns!")
    .send()
    .await?;

println!("Published message: {:?}", rsp);

Ok(())
}
```

- Para obter detalhes da API, consulte [Publish](#) na Referência da API do AWS SDK for Rust.

## SAP ABAP

### SDK para SAP ABAP

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```
TRY.
    oo_result = lo_sns->publish(
testing purposes. "                                " oo_result is returned for
        iv_topicarn = iv_topic_arn
        iv_message = iv_message
    ).
    MESSAGE 'Message published to SNS topic.' TYPE 'I'.
CATCH /aws1/cx_snsnotfoundexception.
    MESSAGE 'Topic does not exist.' TYPE 'E'.
ENDTRY.
```

- Para obter os detalhes da API, consulte [Publicar](#) na Referência da API do AWS SDK para SAP ABAP.

# Publicar mensagens grandes com o Amazon SNS e o Amazon S3

Para publicar mensagens grandes do Amazon SNS, você pode usar a [Biblioteca do cliente em versão ampliada para Java do Amazon SNS](#) ou a [Biblioteca do cliente em versão ampliada para Python do Amazon SNS](#). Essas bibliotecas são úteis para mensagens maiores do que o máximo atual de 256 KB, até o máximo de 2 GB. As bibliotecas salvam a carga útil real em um bucket do Amazon S3 e publicam a referência do objeto armazenado do Amazon S3 no tópico do Amazon SNS. As filas do Amazon SQS inscritas podem usar a [Amazon SQS Extended Client Library for Java](#) para cancelar a referência e recuperar cargas úteis do Amazon S3. Outros endpoints, como o Lambda, podem usar a [Payload Offloading Java Common Library for AWS](#) para cancelar a referência e recuperar a carga útil.

## Note

As bibliotecas de clientes estendidas do Amazon SNS são compatíveis com tópicos padrão e FIFO.

## Tópicos

- [Biblioteca do cliente em versão ampliada para Java](#)
- [Biblioteca do cliente em versão ampliada para Python](#)

## Biblioteca do cliente em versão ampliada para Java

### Tópicos

- [Pré-requisitos](#)
- [Configurar o armazenamento de mensagens](#)
- [Exemplo: publicação de mensagens no Amazon SNS com carga útil armazenada no Amazon S3](#)
- [Outros protocolos de endpoint](#)

### Pré-requisitos

Veja a seguir os pré-requisitos para usar a [Biblioteca do cliente em versão ampliada para Java do Amazon SNS](#):

- Um AWS SDK.

O exemplo nesta página usa o AWS Java SDK. Para instalar e configurar o SDK, consulte [Configurar o AWS SDK for Java](#) no Guia do desenvolvedor do AWS SDK for Java .

- E Conta da AWS com as credenciais adequadas.

Para criar uma Conta da AWS, navegue até a [página AWS inicial](#) e escolha Criar uma AWS conta. Siga as instruções.

Para obter informações sobre credenciais, consulte [Configurar AWS credenciais e região para desenvolvimento no Guia](#) do AWS SDK for Java desenvolvedor.

- Java 8 ou posterior.
- A Biblioteca do cliente em versão ampliada para Java (também disponível no [Maven](#)).

## Configurar o armazenamento de mensagens

A biblioteca Amazon SNS Extended Client usa a biblioteca comum Java Payload Offloading AWS para armazenamento e recuperação de mensagens. Você pode configurar as seguintes [opções de armazenamento de mensagens](#) do Amazon S3:

- Limite de tamanhos de mensagens personalizadas: as mensagens com cargas úteis e atributos que excederem esse tamanho serão automaticamente armazenadas no Amazon S3.
- Sinalizador `alwaysThroughS3`: defina esse valor como `true` para forçar todas as cargas úteis de mensagens a serem armazenadas no Amazon S3. Por exemplo:

```
SNSExtendedClientConfiguration snsExtendedClientConfiguration = new
SNSExtendedClientConfiguration() .withPayloadSupportEnabled(s3Client,
    BUCKET_NAME).withAlwaysThroughS3(true);
```

- Chave KMS personalizada: a chave a ser usada para criptografia do lado do servidor em seu bucket do Amazon S3.
- Nome do bucket: o nome do bucket do Amazon S3 para armazenar cargas úteis de mensagens.

## Exemplo: publicação de mensagens no Amazon SNS com carga útil armazenada no Amazon S3

O código de exemplo a seguir mostra como:

- Criar um tópico de exemplo e uma fila.

- Inscrever a fila para receber mensagens do tópico.
- Publicar uma mensagem de teste.

A carga útil da mensagem é armazenada no Amazon S3 e a referência a ela é publicada. O Amazon SQS Extended Client é usado para receber a mensagem.

## SDK para Java 1.x

### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

Para publicar uma mensagem grande, use a Amazon SNS Extended Client Library for Java. A mensagem que você envia faz referência a um objeto do Amazon S3 que contém o conteúdo real da mensagem.

```
import com.amazon.sqs.javamessaging.AmazonSQSExtendedClient;
import com.amazon.sqs.javamessaging.ExtendedClientConfiguration;
import com.amazonaws.regions.Region;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.sns.AmazonSNS;
import com.amazonaws.services.sns.AmazonSNSClientBuilder;
import com.amazonaws.services.sns.model.CreateTopicRequest;
import com.amazonaws.services.sns.model.PublishRequest;
import com.amazonaws.services.sns.model.SetSubscriptionAttributesRequest;
import com.amazonaws.services.sns.util.Topics;
import com.amazonaws.services.sqs.AmazonSQS;
import com.amazonaws.services.sqs.AmazonSQSClientBuilder;
import com.amazonaws.services.sqs.model.CreateQueueRequest;
import com.amazonaws.services.sqs.model.ReceiveMessageResult;
import software.amazon.sns.AmazonSNSExtendedClient;
import software.amazon.sns.SNSExtendedClientConfiguration;

public class Example {

    public static void main(String[] args) {
        final String BUCKET_NAME = "extended-client-bucket";
```

```
final String TOPIC_NAME = "extended-client-topic";
final String QUEUE_NAME = "extended-client-queue";
final Regions region = Regions.DEFAULT_REGION;

// Message threshold controls the maximum message size that will be
allowed to
// be published
// through SNS using the extended client. Payload of messages
exceeding this
// value will be stored in
// S3. The default value of this parameter is 256 KB which is the
maximum
// message size in SNS (and SQS).
final int EXTENDED_STORAGE_MESSAGE_SIZE_THRESHOLD = 32;

// Initialize SNS, SQS and S3 clients
final AmazonSNS snsClient =
AmazonSNSClientBuilder.standard().withRegion(region).build();
final AmazonSQS sqsClient =
AmazonSQSClientBuilder.standard().withRegion(region).build();
final AmazonS3 s3Client =
AmazonS3ClientBuilder.standard().withRegion(region).build();

// Create bucket, topic, queue and subscription
s3Client.createBucket(BUCKET_NAME);
final String topicArn = snsClient.createTopic(
    new
CreateTopicRequest().withName(TOPIC_NAME)).getTopicArn();
final String queueUrl = sqsClient.createQueue(
    new
CreateQueueRequest().withQueueName(QUEUE_NAME)).getQueueUrl();
final String subscriptionArn = Topics.subscribeQueue(
    snsClient, sqsClient, topicArn, queueUrl);

// To read message content stored in S3 transparently through SQS
extended
// client,
// set the RawMessageDelivery subscription attribute to TRUE
final SetSubscriptionAttributesRequest subscriptionAttributesRequest
= new SetSubscriptionAttributesRequest();
subscriptionAttributesRequest.setSubscriptionArn(subscriptionArn);

subscriptionAttributesRequest.setAttributeName("RawMessageDelivery");
subscriptionAttributesRequest.setAttributeValue("TRUE");
```

```
snsClient.setSubscriptionAttributes(subscriptionAttributesRequest);

// Initialize SNS extended client
// PayloadSizeThreshold triggers message content storage in S3 when
the
// threshold is exceeded
// To store all messages content in S3, use AlwaysThroughS3 flag
final SNSExtendedClientConfiguration snsExtendedClientConfiguration
= new SNSExtendedClientConfiguration()
    .withPayloadSupportEnabled(s3Client, BUCKET_NAME)

.withPayloadSizeThreshold(EXTENDED_STORAGE_MESSAGE_SIZE_THRESHOLD);
final AmazonSNSExtendedClient snsExtendedClient = new
AmazonSNSExtendedClient(snsClient,
    snsExtendedClientConfiguration);

// Publish message via SNS with storage in S3
final String message = "This message is stored in S3 as it exceeds
the threshold of 32 bytes set above.";
snsExtendedClient.publish(topicArn, message);

// Initialize SQS extended client
final ExtendedClientConfiguration sqsExtendedClientConfiguration =
new ExtendedClientConfiguration()
    .withPayloadSupportEnabled(s3Client, BUCKET_NAME);
final AmazonSQSExtendedClient sqsExtendedClient = new
AmazonSQSExtendedClient(sqsClient,
    sqsExtendedClientConfiguration);

// Read the message from the queue
final ReceiveMessageResult result =
sqsExtendedClient.receiveMessage(queueUrl);
System.out.println("Received message is " +
result.getMessages().get(0).getBody());
    }
}
```

## Outros protocolos de endpoint

Tanto as bibliotecas do Amazon SNS quanto do Amazon SQS usam a [Payload Offloading Java Common Library for AWS](#) para armazenar e recuperar cargas úteis de mensagens com o Amazon

S3. Qualquer endpoint habilitado para Java (por exemplo, um endpoint HTTPS implementado em Java) pode usar a mesma biblioteca para cancelar a referência ao conteúdo da mensagem.

Os endpoints que não podem usar a biblioteca comum Java para descarregamento de carga ainda AWS podem publicar mensagens com cargas armazenadas no Amazon S3. Veja a seguir um exemplo de uma referência do Amazon S3 que é publicada pelo exemplo de código acima:

```
[
  "software.amazon.payloadoffloading.PayloadS3Pointer",
  {
    "s3BucketName": "extended-client-bucket",
    "s3Key": "xxxx-xxxxxx-xxxxxx-xxxxxx"
  }
]
```

## Biblioteca do cliente em versão ampliada para Python

### Tópicos

- [Pré-requisitos](#)
- [Configurar o armazenamento de mensagens](#)
- [Exemplo: publicação de mensagens no Amazon SNS com a carga útil armazenada no Amazon S3](#)

### Pré-requisitos

Veja a seguir os pré-requisitos para usar a [Biblioteca do cliente em versão ampliada para Python do Amazon SNS](#):

- Um AWS SDK.

O exemplo nesta página usa o AWS Python SDK Boto3. Para instalar e configurar o SDK, consulte a documentação do [AWS SDK para Python](#).

- E Conta da AWS com as credenciais adequadas.

Para criar uma Conta da AWS, navegue até a [página AWS inicial](#) e escolha Criar uma AWS conta. Siga as instruções.

Para obter informações sobre credenciais, consulte [Credenciais](#) no Guia do desenvolvedor do AWS SDK para Python.

- Python 3.x (ou posterior) e pip.

- A Biblioteca do cliente em versão ampliada para Python (também disponível no [PyPI](#)).

## Configurar o armazenamento de mensagens

Os atributos abaixo estão disponíveis no Boto3 Amazon [SNS](#) Client, Topic [PlatformEndpoint](#) objetos para configurar as opções de armazenamento de mensagens do Amazon S3.

- `large_payload_support`: o nome do bucket do Amazon S3 para armazenar mensagens grandes.
- `message_size_threshold`: o limite para armazenar a mensagem no bucket de mensagens grandes. Deve estar entre 0 e 262144. O padrão é 262144.
- `always_through_s3`: se `True`, então todas as mensagens serão armazenadas no Amazon S3. O padrão é `False`.
- `s3`: o objeto `resource` do Amazon S3 do Boto3 usado para armazenar objetos no Amazon S3. Use isso se quiser controlar o recurso do Amazon S3 (por exemplo, uma configuração ou credenciais personalizadas do Amazon S3). Se não tiver sido definido anteriormente na primeira utilização, o padrão será `boto3.resource("s3")`.

## Exemplo: publicação de mensagens no Amazon SNS com a carga útil armazenada no Amazon S3

O código de exemplo a seguir mostra como:

- Crie um exemplo de tópico do Amazon SNS e uma fila do Amazon SQS.
- Inscrever a fila para receber mensagens do tópico.
- Publicar uma mensagem de teste.
- A carga útil da mensagem é armazenada no Amazon S3 e a referência a ela é publicada.
- Imprima a mensagem publicada da fila junto com a mensagem original recuperada do Amazon S3.

Para publicar uma mensagem grande, use a Biblioteca do cliente em versão ampliada para Python. A mensagem que você envia faz referência a um objeto do Amazon S3 que contém o conteúdo real da mensagem.

```
import boto3
import sns_extended_client
from json import loads
```

```
s3_extended_payload_bucket = "extended-client-bucket-store"
TOPIC_NAME = "---TOPIC-NAME---"
QUEUE_NAME = "---QUEUE-NAME---"

# Create an helper to fetch message from S3
def get_msg_from_s3(body):
    json_msg = loads(body)
    s3_client = boto3.client("s3")
    s3_object = s3_client.get_object(
        Bucket=json_msg[1].get("s3BucketName"), Key=json_msg[1].get("s3Key")
    )
    msg = s3_object.get("Body").read().decode()
    return msg

# Create an helper to fetch and print message SQS queue and S3
def fetch_and_print_from_sqs(sqs, queue_url):
    """Handy Helper to fetch and print message from SQS queue and S3"""
    message = sqs.receive_message(
        QueueUrl=queue_url, MessageAttributeNames=["All"], MaxNumberOfMessages=1
    ).get("Messages")[0]
    message_body = message.get("Body")
    print("Published Message: {}".format(message_body))
    print("Message Stored in S3 Bucket is: {}\n".format(get_msg_from_s3(message_body)))

# Initialize the SNS client and create SNS Topic
sns_extended_client = boto3.client("sns", region_name="us-east-1")
create_topic_response = sns_extended_client.create_topic(Name=TOPIC_NAME)
demo_topic_arn = create_topic_response.get("TopicArn")

# Create and subscribe an SQS queue to the SNS client
sqs = boto3.client("sqs")
demo_queue_url = sqs.create_queue(QueueName=QUEUE_NAME).get("QueueUrl")
demo_queue_arn = sqs.get_queue_attributes(QueueUrl=demo_queue_url,
AttributeNames=["QueueArn"])[ "Attributes" ].get("QueueArn")
# Set the RawMessageDelivery subscription attribute to TRUE
sns_extended_client.subscribe(TopicArn=demo_topic_arn, Protocol="sqs",
Endpoint=demo_queue_arn, Attributes={"RawMessageDelivery":"true"})

sns_extended_client.large_payload_support = s3_extended_payload_bucket

# To store all messages content in S3, set always_through_s3 to True
# In the example, we set message size threshold as 32 bytes, adjust this threshold as
per your usecase
```

```
# Message will only be uploaded to S3 when its payload size exceeded threshold
sns_extended_client.message_size_threshold = 32
sns_extended_client.publish(
    TopicArn=demo_topic_arn,
    Message="This message should be published to S3 as it exceeds the
message_size_threshold limit",
)
# Print message stored in s3
fetch_and_print_from_sqs(sqs, demo_queue_url)
```

## Saída

```
Published Message:
[
  "software.amazon.payloadoffloading.PayloadS3Pointer",
  {
    "s3BucketName": "extended-client-bucket-store",
    "s3Key": "xxxx-xxxxx-xxxxx-xxxxxx"
  }
]
Message Stored in S3 Bucket is: This message should be published to S3 as it exceeds
the message_size_threshold limit
```

## Atributos de mensagem do Amazon SNS

O Amazon SNS oferece suporte à entrega de atributos de mensagem, o que permite que você forneça itens de metadados estruturados (como time stamps, dados geoespaciais, assinaturas e identificadores) sobre a mensagem. Para assinaturas SQS, no máximo dez atributos de mensagem podem ser enviados quando [Raw Message Delivery](#) (Entrega de mensagens brutas) estiver habilitado. Para enviar mais de dez atributos de mensagem, é necessário desabilitar Raw Message Delivery (Entrega de mensagens brutas). As mensagens com mais de 10 atributos de mensagem direcionadas para assinaturas do Amazon SQS habilitadas para entrega bruta de mensagens serão descartadas como erros do lado do cliente.

Os atributos de mensagem são opcionais e separados do corpo da mensagem, mas são enviados junto com ele. O destinatário pode usar essas informações para decidir como lidar com ela sem ter de primeiro processar o corpo da mensagem.

Para obter informações sobre como enviar mensagens com atributos usando o AWS Management Console ou o AWS SDK for Java, veja o tutorial [Para publicar mensagens nos tópicos do Amazon SNS usando o AWS Management Console](#).

 Note

Os atributos de mensagens são enviados somente quando a estrutura da mensagem é String, e não JSON.

Também é possível usar atributos de mensagem para ajudar a estruturar a mensagem de notificação por push para endpoints móveis. Nesse cenário, os atributos de mensagens são usados somente para ajudar a estruturar a mensagem de notificação por push. Os atributos não são entregues ao endpoint do mesmo modo que são ao enviar mensagens com atributos de mensagem para endpoints do Amazon SQS.

Você também pode usar atributos de mensagens para que suas mensagens possam ser filtradas usando políticas de filtro de assinatura. Você pode aplicar as políticas de filtro a inscrições em tópicos. Quando uma política de filtro é aplicada com o escopo de políticas de filtro definido como `MessageAttributes` (padrão), uma assinatura recebe somente aquelas mensagens que têm atributos aceitos pela política. Para obter mais informações, consulte [Filtragem de mensagens do Amazon SNS](#).

 Note

Quando atributos de mensagem são usados para filtragem, o valor deve ser uma string JSON válida. Isso garante que a mensagem seja entregue a uma assinatura com a filtragem de atributos de mensagem ativada.

## Validação e itens de atributos de mensagem

Cada atributo de mensagem consiste nos seguintes itens:

- Nome: o nome do atributo de mensagem pode conter os seguintes caracteres: A-Z, a-z, 0-9, sublinhado (`_`), hífen (`-`) e ponto (`.`). O nome não deve iniciar ou terminar com um ponto, e nem ter pontos sucessivos. O nome diferencia maiúsculas e minúsculas e deve ser exclusivo entre todos os nomes de atributos para a mensagem. O nome pode ter até 256 caracteres. O nome não pode

começar com `AWS.` ou `Amazon.` (ou quaisquer variações em maiúsculas e minúsculas) porque esses prefixos são reservados para uso da Amazon Web Services.

- **Tipo:** os tipos de dados de atributo de mensagem compatíveis são `String`, `String.Array`, `Number` e `Binary`. O tipo de dados tem as mesmas restrições no conteúdo que o corpo da mensagem. O tipo de dados diferencia maiúsculas de minúsculas e pode ter até 256 bytes. Para obter mais informações, consulte a seção [Tipos de dados de atributo de mensagem e validação](#).
- **Valor:** o valor do atributo de mensagem especificado pelo usuário. Para tipos de dados de string, o atributo de valor tem as mesmas restrições no conteúdo que o corpo da mensagem. Para obter mais informações, consulte a ação [Publish](#) na Referência da API do Amazon Simple Notification Service.

Nome, tipo e valor não podem estar vazios ou nulos. Além disso, o corpo da mensagem não pode estar vazio ou nulo. Todas as partes do atributo da mensagem, incluindo o nome, o tipo e o valor, são incluídas na restrição de tamanho da mensagem, que é de 256 KB.

## Tipos de dados de atributo de mensagem e validação

Os tipos de dados de atributo de mensagem identificam como os valores de atributo de mensagem são processados pelo Amazon SNS. Por exemplo, se o tipo for um número, o Amazon SNS validará que é um número.

O Amazon SNS é compatível com os seguintes tipos de dados lógicos em todos os endpoints, com exceção daqueles com observação contrária:

- **String:** as strings são Unicode com codificação binária UTF-8. Para obter uma lista de valores de código, consulte [http://en.wikipedia.org/wiki/ASCII#ASCII\\_printable\\_characters](http://en.wikipedia.org/wiki/ASCII#ASCII_printable_characters).

### Note

Valores substitutos não são aceitos nos atributos da mensagem. Por exemplo, usar um valor substituto para representar um emoji apresentará o seguinte erro: `Invalid attribute value was passed in for message attribute`.

- **String.Array:** uma matriz, formatada como uma string, que pode conter vários valores. Os valores podem ser strings, números ou as palavras-chave `true`, `false` e `null`. Um `String.Array` de número ou tipo booleano não precisa de aspas. Vários valores `String.Array` são separados por vírgulas.

Esse tipo de dados não é compatível com as assinaturas do AWS Lambda. Se você especificar esse tipo de dados para endpoints do Lambda, ele será passado como o tipo de dados `String` na carga útil JSON que o Amazon SNS entrega ao Lambda.

- **Number:** números são inteiros positivos ou negativos ou com ponto flutuante. Números têm alcance e precisão suficientes para abranger a maioria dos possíveis valores que números inteiros, flutuantes e duplicados normalmente comportam. Um número pode ter um valor de  $-10^9$  a  $10^9$ , com 5 dígitos de precisão após o ponto decimal. Zeros iniciais e finais são cortados.

Esse tipo de dados não é compatível com as assinaturas do AWS Lambda. Se você especificar esse tipo de dados para endpoints do Lambda, ele será passado como o tipo de dados `String` na carga útil JSON que o Amazon SNS entrega ao Lambda.

- **Binary:** os atributos de tipo binário podem armazenar qualquer dado binário, por exemplo, dados compactados, dados criptografados ou imagens.

## Atributos de mensagens reservados para notificações móveis por push

A tabela a seguir lista os atributos de mensagens reservadas para serviços de notificação móvel por push que você pode usar para estruturar a mensagem de notificação por push:

Serviço de notificação por push	Atributo de mensagem reservada
ADM	<code>AWS.SNS.MOBILE.ADM.TTL</code>
APNs <sup>1</sup>	<code>AWS.SNS.MOBILE.APNS_MDM.TTL</code>
	<code>AWS.SNS.MOBILE.APNS_MDM_SANDBOX.TTL</code>
	<code>AWS.SNS.MOBILE.APNS_PASSBOOK.TTL</code>
	<code>AWS.SNS.MOBILE.APNS_PASSBOOK_SANDBOX.TTL</code>
	<code>AWS.SNS.MOBILE.APNS_SANDBOX.TTL</code>
	<code>AWS.SNS.MOBILE.APNS_VOIP.TTL</code>
	<code>AWS.SNS.MOBILE.APNS_VOIP_SANDBOX.TTL</code>

Serviço de notificação por push	Atributo de mensagem reservada
	<code>AWS.SNS.MOBILE.APNS.COLLAPSE_ID</code>
	<code>AWS.SNS.MOBILE.APNS.PRIORITY</code>
	<code>AWS.SNS.MOBILE.APNS.PUSH_TYPE</code>
	<code>AWS.SNS.MOBILE.APNS.TOPIC</code>
	<code>AWS.SNS.MOBILE.APNS.TTL</code>
Baidu	<code>AWS.SNS.MOBILE.BAIDU.DeployStatus</code>
	<code>AWS.SNS.MOBILE.BAIDU.MessageKey</code>
	<code>AWS.SNS.MOBILE.BAIDU.MessageType</code>
	<code>AWS.SNS.MOBILE.BAIDU.TTL</code>
FCM	<code>AWS.SNS.MOBILE.FCM.TTL</code>
	<code>AWS.SNS.MOBILE.GCM.TTL</code>
macOS	<code>AWS.SNS.MOBILE.MACOS_SANDBOX.TTL</code>
	<code>AWS.SNS.MOBILE.MACOS.TTL</code>
MPNS	<code>AWS.SNS.MOBILE.MPNS.NotificationClass</code>
	<code>AWS.SNS.MOBILE.MPNS.TTL</code>
	<code>AWS.SNS.MOBILE.MPNS.Type</code>
WNS	<code>AWS.SNS.MOBILE.WNS.CachePolicy</code>
	<code>AWS.SNS.MOBILE.WNS.Group</code>
	<code>AWS.SNS.MOBILE.WNS.Match</code>
	<code>AWS.SNS.MOBILE.WNS.SuppressPopup</code>

Serviço de notificação por push	Atributo de mensagem reservada
	<code>AWS.SNS.MOBILE.WNS.Tag</code>
	<code>AWS.SNS.MOBILE.WNS.TTL</code>
	<code>AWS.SNS.MOBILE.WNS.Type</code>

<sup>1</sup> A Apple rejeitará as notificações do Amazon SNS se os atributos das mensagens não atenderem aos requisitos. Para obter mais detalhes, consulte [Enviar solicitações de notificação para APNs](#) no site do desenvolvedor da Apple.

## Agrupamento de mensagens do Amazon SNS em lotes

### O que é o agrupamento de mensagens em lotes?

Uma alternativa à publicação de mensagens em tópicos padrão ou FIFO em solicitações individuais da API `Publish` é usar a API `PublishBatch` do Amazon SNS para publicar até 10 mensagens em uma única solicitação de API. O envio de mensagens em lotes pode ajudar a reduzir os custos associados à conexão de aplicações distribuídas ([sistema de mensagens A2A](#)) ou ao envio de notificações a pessoas ([sistema de mensagens A2P](#)) com o Amazon SNS em um fator de até 10. O Amazon SNS tem cotas que definem quantas mensagens é possível publicar em um tópico por segundo com base na região de operação. Consulte a página [Endpoints e cotas do Amazon SNS](#) no guia Referência geral da AWS para obter mais informações sobre cotas de API.

#### Note

O tamanho total agregado de todas as mensagens que você envia em uma única solicitação da API `PublishBatch` não pode exceder 262.144 bytes (256 KB).  
A API `PublishBatch` usa a mesma ação da API `Publish` para políticas do IAM.

### Como funciona o agrupamento de mensagens em lotes?

A publicação de mensagens com a API `PublishBatch` é semelhante à publicação de mensagens com a API `Publish`. A principal diferença é que cada mensagem em uma solicitação da API

`PublishBatch` precisa ser atribuída a um ID de lote exclusivo com até 80 caracteres. Dessa maneira, podem ser retornadas respostas individuais da API pelo Amazon SNS para cada mensagem em um lote para confirmar se cada mensagem foi publicada ou se ocorreu uma falha. Para mensagens que estão sendo publicadas em tópicos FIFO, além de incluir a atribuição de um ID de lote exclusivo, ainda é necessário incluir um `MessageDeduplicationID` e `MessageGroupId` para cada mensagem individual.

## Exemplos

Publicação de um lote de 10 mensagens em um tópico padrão

```
// Imports
import com.amazonaws.services.sns.AmazonSNS;
import com.amazonaws.services.sns.model.PublishBatchRequest;
import com.amazonaws.services.sns.model.PublishBatchRequestEntry;
import com.amazonaws.services.sns.model.PublishBatchResult;
import com.amazonaws.services.sns.model.AmazonSNSException;
import java.util.List;
import java.util.stream.Collectors;

// Code
private static final int MAX_BATCH_SIZE = 10;

public static void publishBatchToTopic(AmazonSNS snsClient, String topicArn) {
    try {
        // Create the batch entries to send
        List<PublishBatchRequestEntry> entries = IntStream.range(0, MAX_BATCH_SIZE)
            .mapToObj(i -> new PublishBatchRequestEntry()
                .withId("id" + i)
                .withMessage("message" + i))
            .collect(Collectors.toList());

        // Create the batch request
        PublishBatchRequest request = new PublishBatchRequest()
            .withTopicArn(topicArn)
            .withPublishBatchRequestEntries(entries);

        // Publish the batch request
        PublishBatchResult publishBatchResult = snsClient.publishBatch(request);

        // Handle the successfully sent messages
    }
}
```

```
        publishBatchResult.getSuccessful().forEach(publishBatchResultEntry -> {
            System.out.println("Batch Id for successful message: " +
publishBatchResultEntry.getId());
            System.out.println("Message Id for successful message: " +
publishBatchResultEntry.getMessageId());
        });

        // Handle the failed messages
        publishBatchResult.getFailed().forEach(batchResultErrorEntry -> {
            System.out.println("Batch Id for failed message: " +
batchResultErrorEntry.getId());
            System.out.println("Error Code for failed message: " +
batchResultErrorEntry.getCode());
            System.out.println("Sender Fault for failed message: " +
batchResultErrorEntry.getSenderFault());
            System.out.println("Failure Message for failed message: " +
batchResultErrorEntry.getMessage());
        });
    } catch (AmazonSNSException e) {
        // Handle any exceptions from the request
        System.err.println(e.getMessage());
        System.exit(1);
    }
}
```

## Publicação de um lote de 10 mensagens em um tópico FIFO

```
// Imports
import com.amazonaws.services.sns.AmazonSNS;
import com.amazonaws.services.sns.model.PublishBatchRequest;
import com.amazonaws.services.sns.model.PublishBatchRequestEntry;
import com.amazonaws.services.sns.model.PublishBatchResult;
import com.amazonaws.services.sns.model.AmazonSNSException;
import java.util.List;
import java.util.stream.Collectors;

// Code
private static final int MAX_BATCH_SIZE = 10;

public static void publishBatchToFifoTopic(AmazonSNS snsClient, String topicArn) {
    try {
```

```
// Create the batch entries to send
List<PublishBatchRequestEntry> entries = IntStream.range(0, MAX_BATCH_SIZE)
    .mapToObj(i -> new PublishBatchRequestEntry()
        .withId("id" + i)
        .withMessage("message" + i)
        .withMessageGroupId("groupId")
        .withMessageDeduplicationId("deduplicationId" + i))
    .collect(Collectors.toList());

// Create the batch request
PublishBatchRequest request = new PublishBatchRequest()
    .withTopicArn(topicArn)
    .withPublishBatchRequestEntries(entries);

// Publish the batch request
PublishBatchResult publishBatchResult = snsClient.publishBatch(request);

// Handle the successfully sent messages
publishBatchResult.getSuccessful().forEach(publishBatchResultEntry -> {
    System.out.println("Batch Id for successful message: " +
publishBatchResultEntry.getId());
    System.out.println("Message Id for successful message: " +
publishBatchResultEntry.getMessageId());
    System.out.println("SequenceNumber for successful message: " +
publishBatchResultEntry.getSequenceNumber());
});

// Handle the failed messages
publishBatchResult.getFailed().forEach(batchResultErrorEntry -> {
    System.out.println("Batch Id for failed message: " +
batchResultErrorEntry.getId());
    System.out.println("Error Code for failed message: " +
batchResultErrorEntry.getCode());
    System.out.println("Sender Fault for failed message: " +
batchResultErrorEntry.getSenderFault());
    System.out.println("Failure Message for failed message: " +
batchResultErrorEntry.getMessage());
});

} catch (AmazonSNSException e) {
    // Handle any exceptions from the request
    System.err.println(e.getMessage());
    System.exit(1);
}
```

```
}
```

# Filtragem de mensagens do Amazon SNS

Por padrão, um assinante de tópico do Amazon SNS recebe todas as mensagens publicadas no tópico. Para receber apenas um subconjunto de mensagens, um assinante deve atribuir uma política de filtro à assinatura do tópico.

Uma política de filtro é um objeto JSON que contém propriedades que definem quais mensagens o assinante recebe. O Amazon SNS oferece suporte a políticas que atuam nos atributos da mensagem ou no corpo da mensagem, de acordo com o escopo da política de filtro que você definiu para a assinatura. As políticas de filtro para o corpo da mensagem pressupõem que a carga útil da mensagem seja um objeto JSON bem formado.

Se uma assinatura não tiver uma política de filtro, o assinante receberá todas as mensagens publicadas ao tópico. Quando você publica uma mensagem em um tópico com uma política de filtro em vigor, o Amazon SNS compara os atributos da mensagem ou o corpo da mensagem com as propriedades na política de filtro para cada assinatura do tópico. Se houver correspondência com qualquer uma das propriedades dos atributos ou do corpo da mensagem, o Amazon SNS enviará a mensagem para o assinante. Caso contrário, o Amazon SNS não enviará a mensagem para esse assinante.

Para obter mais informações, consulte [Filtre mensagens publicadas em tópicos](#).

## Tópicos

- [Escopo de política de filtro de assinaturas do Amazon SNS](#)
- [Políticas de filtro de assinatura do Amazon SNS](#)
- [Aplicar uma política de filtro de assinatura](#)
- [Remover uma política de filtro de assinatura](#)

## Escopo de política de filtro de assinaturas do Amazon SNS

O atributo de assinatura `FilterPolicyScope` permite que você escolha o escopo da filtragem definindo um dos seguintes valores:

- `MessageAttributes`: a política de filtro é aplicada aos atributos da mensagem. Esse é o padrão.
- `MessageBody`: a política de filtro é aplicada ao corpo da mensagem.

**Note**

Se nenhum escopo de política de filtro for definido para uma política de filtro existente, o escopo assumirá `MessageAttributes` como padrão.

## Políticas de filtro de assinatura do Amazon SNS

Uma política de filtro de assinatura permite especificar nomes de propriedades e atribuir uma lista de valores para cada nome de propriedade. Para ter mais informações, consulte [Filtragem de mensagens do Amazon SNS](#).

Quando o Amazon SNS avalia os atributos ou o corpo da mensagem em relação à política de filtro de assinatura, ele ignora atributos que não estão especificados na política.

**Important**

AWS serviços como IAM e Amazon SNS usam um modelo de computação distribuída chamado consistência eventual. As adições ou alterações a uma política de filtro de assinatura podem levar até 15 minutos para entrarem em vigor.

Uma assinatura aceita uma mensagem nas seguintes condições:

- Quando o escopo da política de filtro está definido como `MessageAttributes`, cada nome de propriedade na política de filtro corresponde ao nome de um atributo da mensagem. Para cada nome de propriedade correspondente na política de filtro, pelo menos um valor de propriedade corresponde ao valor do atributo da mensagem.
- Quando o escopo da política de filtro está definido como `MessageBody`, cada nome de propriedade na política de filtro corresponde ao nome de uma propriedade do corpo da mensagem. Para cada nome de propriedade correspondente na política de filtro, pelo menos um valor de propriedade corresponde ao valor da propriedade do corpo da mensagem.

Atualmente, o Amazon SNS é compatível com os seguintes operadores de filtro:

- [Lógica AND](#)
- [Lógica OR](#)

- [Operador OR](#)
- [Correspondência de chaves](#)
- [Correspondência exata do valor numérico](#)
- [Correspondência tudo-exceto de valor numérico](#)
- [Correspondência de valores numéricos](#)
- [Correspondência de valores de string](#)
- [Correspondência tudo-exceto de valor de string](#)
- [Correspondência de string usando um prefixo com o operador tudo-exceto](#)
- [Correspondência de igual-ignorar-capitalização do valor da string](#)
- [Correspondência do endereço IP do valor da string](#)
- [Correspondência de prefixo de valores de string](#)
- [Correspondência de sufixo de valores de string](#)

## Exemplo de políticas de filtro

O exemplo a seguir mostra uma carga útil da mensagem entregue por um tópico do Amazon SNS que processa as transações de clientes.

O primeiro exemplo inclui o campo `MessageAttributes` com atributos que descrevem a transação:

- Interesses do cliente
- Nome do repositório
- Estado do evento
- Preço de compra em USD

Como essa mensagem inclui o campo `MessageAttributes`, qualquer assinatura de tópico que defina uma `FilterPolicy` pode aceitar ou rejeitar seletivamente a mensagem, desde que `FilterPolicyScope` esteja definido como `MessageAttributes` na assinatura. Para obter mais informações sobre a aplicação de atributos a uma mensagem, consulte [Atributos de mensagem do Amazon SNS](#).

```
{
```

```

    "Type": "Notification",
    "MessageId": "a1b2c34d-567e-8f90-g1h2-i345j67klmn8",
    "TopicArn": "arn:aws:sns:us-east-2:123456789012:MyTopic",
    "Message": "message-body-with-transaction-details",
    "Timestamp": "2019-11-03T23:28:01.631Z",
    "SignatureVersion": "4",
    "Signature": "signature",
    "UnsubscribeURL": "unsubscribe-url",
    "MessageAttributes": {
      "customer_interests": {
        "Type": "String.Array",
        "Value": "[\"soccer\", \"rugby\", \"hockey\"]"
      },
      "store": {
        "Type": "String",
        "Value": "example_corp"
      },
      "event": {
        "Type": "String",
        "Value": "order_placed"
      },
      "price_usd": {
        "Type": "Number",
        "Value": "210.75"
      }
    }
  }
}

```

O exemplo a seguir mostra os mesmos atributos incluídos no campo Message, também chamado de carga útil da mensagem ou corpo da mensagem. Qualquer assinatura de tópico que inclui uma FilterPolicy pode aceitar ou rejeitar seletivamente a mensagem, desde que FilterPolicyScope esteja definido como MessageBody na assinatura.

```

{
  "Type": "Notification",
  "MessageId": "a1b2c34d-567e-8f90-g1h2-i345j67klmn8",
  "TopicArn": "arn:aws:sns:us-east-2:123456789012:MyTopic",
  "Message": "{
    \"customer_interests\": [\"soccer\", \"rugby\", \"hockey\"],
    \"store\": \"example_corp\",
    \"event\": \"order_placed\",
    \"price_usd\": 210.75
  }",
}

```

```
"Timestamp": "2019-11-03T23:28:01.631Z",
"SignatureVersion": "4",
"Signature": "signature",
"UnsubscribeURL": "unsubscribe-url"
}
```

As políticas de filtro a seguir aceitam ou rejeitam mensagens com base em seus nomes e valores de propriedades.

## Uma política que aceita a mensagem de exemplo

As propriedades na política de filtro de assinatura a seguir correspondem aos atributos da mensagem de exemplo. Observe que a mesma política de filtro funciona para um `FilterPolicyScope` definido como `MessageAttributes` ou `MessageBody`. Cada assinante escolhe seu escopo de filtragem de acordo com a composição das mensagens que recebe do tópico.

Se qualquer propriedade única nessa política não corresponder a um atributo da mensagem, a política rejeitará a mensagem.

```
{
  "store": ["example_corp"],
  "event": [{"anything-but": "order_cancelled"}],
  "customer_interests": [
    "rugby",
    "football",
    "baseball"
  ],
  "price_usd": [{"numeric": [">=", 100]}]
}
```

## Uma política que rejeita a mensagem de exemplo

A política de filtro de assinatura a seguir tem várias incompatibilidades entre seus atributos e as propriedades da mensagem de exemplo. Por exemplo, como o nome da propriedade `encrypted` não está presente nos atributos da mensagem, essa propriedade de política faz com que a mensagem seja rejeitada, independentemente do valor atribuído a ela.

Se ocorrer qualquer incompatibilidade, a política rejeitará a mensagem.

```
{
  "store": ["example_corp"],
```

```
"event": ["order_cancelled"],
"encrypted": [false],
"customer_interests": [
  "basketball",
  "baseball"
]
}
```

## Restrições da política de filtro

Quando você cria uma política de filtro para uma assinatura do Amazon SNS, é importante entender como as chaves na política são contadas. Os principais aspectos a serem lembrados são:

1. Chaves principais — As chaves principais são as chaves de nível superior na política de filtro. Essas são as chaves para as quais você especifica valores ou restrições.
2. Nomes de atributos — As chaves principais são consideradas os nomes dos atributos na política de filtro. Os valores ou restrições que você especificar para essas chaves serão aplicados aos atributos correspondentes na carga útil da mensagem.
3. Valores válidos — Os valores especificados para as chaves principais devem ser uma string, uma matriz de strings ou um número. Se o valor for um objeto (por exemplo, um objeto JSON), ele não será contado como uma chave válida na política de filtro.

Vamos considerar o seguinte exemplo de política de filtro:

```
{
  "state": ["SUCCESS"],
  "severity": [{ "exists": true }],
  "message": [{ "exists": true }],
  "finding": {
    "standard_control": [{ "exists": true }],
    "region": [{ "exists": true }],
    "account": [{ "exists": true }]
  }
}
```

Neste exemplo, as chaves a seguir são contadas como parte da política de filtro:

- `state`
- `severity`

- message
- standard\_control
- region
- account

A descoberta da chave não é contada, pois contém um objeto JSON como valor, em vez de uma string, matriz de strings ou um número.

Outro exemplo:

```
{
  "key_a": {
    "key_b": {
      "key_c": {
        "key_d": ["value_one", "value_two", "value_three", "value_four"]
      }
    },
    "key_e": {
      "key_f": ["value_one", "value_two", "value_three"]
    }
  }
}
```

Nesse caso, somente as chaves 1 key\_d e 3 key\_f são contadas como parte da política de filtro, pois elas têm valores atribuídos a elas que são uma string ou uma matriz de strings. As chaves principais key\_a, key\_b, e não key\_c são contadas, pois contêm objetos JSON aninhados como seus valores.

## Tópicos

- [Restrições da política de filtro](#)
- [Restrições de política para filtragem baseada em atributo](#)
- [Restrições de política para filtragem baseada em carga útil](#)

## Restrições da política de filtro

- Correspondência de strings — Para correspondência de strings na política de filtro, a comparação faz distinção entre maiúsculas e minúsculas.

- Correspondência numérica — Para correspondência numérica, o valor pode variar de  $-10^9$  a  $10^9$  (-1 bilhão a 1 bilhão), com cinco dígitos de precisão após o ponto decimal.
- Complexidade da política de filtro — Para a complexidade da política de filtro, a combinação total de valores não deve exceder 150. Para calcular a combinação total, multiplique o número de valores em cada matriz na política de filtro.

Considere o seguinte exemplo de política:

```
{
  "key_a": ["value_one", "value_two", "value_three"],
  "key_b": ["value_one"],
  "key_c": ["value_one", "value_two"]
}
```

Nesta política:

- A primeira matriz tem 3 valores
- A segunda matriz tem 1 valor
- A terceira matriz tem 2 valores

A combinação total é calculada da seguinte forma:

- $3 \times 1 \times 2 = 6$

### Sintaxe da política de filtro

O JSON da política de filtro pode conter o seguinte:

- Strings entre aspas
- Números
- As palavras-chave `true`, `false` e `null`, sem aspas

Ao usar a API do Amazon SNS, você deve passar o JSON da política de filtro como uma string UTF-8 válida.

### Limites da política de filtro

- O tamanho máximo de uma política de filtro é 256 KB.
- Por padrão, você pode ter até 200 políticas de filtro por tópico e 10.000 políticas de filtro por AWS conta.
- Esse limite de política não impediria que as assinaturas de fila do Amazon SQS fossem criadas com a API `Subscribe`. No entanto, falhará quando você anexar a política de filtro à chamada da API `Subscribe` (ou na chamada da API `SetSubscriptionAttributes`).
- Para aumentar essa cota, use [AWS Service Quotas](#).

## Restrições de política para filtragem baseada em atributo

- A filtragem baseada em atributo é a opção padrão. `FilterPolicyScope` é definido como `MessageAttributes` na assinatura.
- O Amazon SNS não aceita uma política de filtro aninhada para filtragem baseada em atributo.
- O Amazon SNS compara as propriedades de política somente com atributos de mensagens que tenham os seguintes tipos de dados:
  - `String`
  - `String.Array`

### Important

Transmitir objetos em matrizes não é recomendado porque pode gerar resultados inesperados devido ao aninhamento, o que não é compatível com a filtragem baseada em atributos. Usar filtragem baseada em carga para políticas aninhadas.

- `Number`
- O Amazon SNS ignora atributos de mensagens com o tipo de dados `Binary`.
- Uma política de filtro pode ter no máximo cinco nomes de atributo.

## Restrições de política para filtragem baseada em carga útil

O Amazon SNS aceita uma política de filtro aninhado para filtragem baseada em carga útil. Para calcular a combinação total de valores na política de filtro, multiplique o número de valores em cada matriz aninhada.

Considere o seguinte exemplo de política:

```
{
  "key_a": {
    "key_b": {
      "key_c": ["value_one", "value_two", "value_three", "value_four"]
    }
  },
  "key_d": {
    "key_e": ["value_one", "value_two", "value_three"]
  }
}
```

Nesta política:

- A primeira matriz tem quatro valores em uma chave aninhada de três níveis.
- O segundo tem três valores em uma chave aninhada de dois níveis.

A combinação total é calculada da seguinte forma:

- $4 \times 3 \times 3 \times 2 = 72$

### Limites da política

Uma política de filtro pode ter no máximo cinco chaves principais (chaves de nível superior). Para uma política aninhada, somente as chaves principais são contabilizadas no limite de cinco chaves.

### Intervalo numérico

Para correspondência numérica na política de filtro, o valor pode variar de  $-10^9$  a  $10^9$  (-1 bilhão a 1 bilhão), com cinco dígitos de precisão após o ponto decimal.

### Mudando para a filtragem baseada em carga útil

Para mudar da filtragem baseada em atributo (padrão) para a filtragem baseada em carga útil, defina `FilterPolicyScope` como `MessageBody` na assinatura.

## Lógica E/OU

Use as operações que incluem lógica E/OU para estabelecer correspondência com propriedades de atributos ou corpos de mensagens.

## Tópicos

- [Lógica AND](#)
- [Lógica OR](#)
- [Operador OR](#)

## Lógica AND

Aplique a lógica E usando vários nomes de propriedade.

Considere a seguinte política:

```
{
  "customer_interests": ["rugby"],
  "price_usd": [{"numeric": [ ">", 100]}]
}
```

Ela estabelece correspondência com qualquer propriedade de atributo ou corpo de mensagem com o valor de `customer_interests` definido como `rugby` e o valor de `price_usd` definido como um número maior que 100.

### Note

Não é possível aplicar a lógica AND aos valores do mesmo atributo.

## Lógica OR

Aplique a lógica OU atribuindo vários valores a um nome de propriedade.

Considere a seguinte política:

```
{
  "customer_interests": ["rugby", "football", "baseball"]
}
```

Ela estabelece correspondência com qualquer propriedade de atributo ou corpo de mensagem com o valor de `customer_interests` definido como `rugby`, `football` ou `baseball`.

## Operador OR

Você pode usar o operador "\$or" para definir explicitamente uma política de filtro para expressar a relação OR entre vários atributos na política.

O Amazon SNS só reconhece uma relação "\$or" quando a política atende a todas as condições a seguir. Quando todas essas condições não são atendidas, "\$or" é tratado como um nome de atributo regular, o mesmo que qualquer outra string na política.

- Há um atributo do campo "\$or" na regra seguido por uma matriz, por exemplo "\$or" : [ ].
- Há pelo menos dois objetos na matriz "\$or": "\$or": [ {}, {} ].
- Nenhum dos objetos na matriz "\$or" tem nomes de campo que sejam palavras-chave reservadas.

Caso contrário, "\$or" é tratado como um nome de atributo normal, o mesmo que outras strings na política.

A política a seguir não é analisada como uma relação OR porque numérico e prefixo são palavras-chave reservadas.

```
{
  "$or": [ {"numeric" : 123}, {"prefix": "abc"} ]
}
```

### Exemplos de operador **OR**

OR padrão:

```
{
  "source": [ "aws.cloudwatch" ],
  "$or": [
    { "metricName": [ "CPUUtilization" ] },
    { "namespace": [ "AWS/EC2" ] }
  ]
}
```

A lógica do filtro para essa política é:

```
"source" && ("metricName" || "namespace")
```

Ele corresponde a um dos seguintes atributos de mensagens:

```
"source": {"Type": "String", "Value": "aws.cloudwatch"},
"metricName": {"Type": "String", "Value": "CPUUtilization"}
```

ou

```
"source": {"Type": "String", "Value": "aws.cloudwatch"},
"namespace": {"Type": "String", "Value": "AWS/EC2"}
```

Também estabelece correspondência com um dos seguintes corpos de mensagem:

```
{
  "source": "aws.cloudwatch",
  "metricName": "CPUUtilization"
}
```

ou

```
{
  "source": "aws.cloudwatch",
  "namespace": "AWS/EC2"
}
```

Restrições de políticas que incluem relações **OR**

Considere a seguinte política:

```
{
  "source": [ "aws.cloudwatch" ],
  "$or": [
    { "metricName": [ "CPUUtilization", "ReadLatency" ] },
    {
      "metricType": [ "MetricType" ] ,
      "$or" : [
        { "metricId": [ 1234, 4321 ] },
        { "spaceId": [ 1000, 2000, 3000 ] }
      ]
    }
  ]
}
```

A lógica dessa política também pode ser simplificada como:

```
("source" AND "metricName")
OR
("source" AND "metricType" AND "metricId")
OR
("source" AND "metricType" AND "spaceId")
```

O cálculo da complexidade para políticas com relacionamentos OR pode ser simplificado como a soma das complexidades da combinação para cada declaração OR.

A combinação total é calculada da seguinte forma:

```
(source * metricName) + (source * metricType * metricId) + (source * metricType *
spaceId)
= (1 * 2) + (1 * 1 * 2) + (1 * 1 * 3)
= 7
```

source tem um valor, metricName tem dois valores, metricType tem um valor, metricId tem dois valores e spaceId tem três valores.

Considere a seguinte política de filtro aninhada:

```
{
  "$or": [
    { "metricName": [ "CPUUtilization", "ReadLatency" ] },
    { "namespace": [ "AWS/EC2", "AWS/ES" ] }
  ],
  "detail" : {
    "scope" : [ "Service" ],
    "$or": [
      { "source": [ "aws.cloudwatch" ] },
      { "type": [ "CloudWatch Alarm State Change" ] }
    ]
  }
}
```

A lógica dessa política pode ser simplificada como:

```
("metricName" AND ("detail"."scope" AND "detail"."source"))
OR
("metricName" AND ("detail"."scope" AND "detail"."type"))
```

```
OR
("namespace" AND ("detail"."scope" AND "detail"."source"))
OR
("namespace" AND ("detail"."scope" AND "detail"."type"))
```

O cálculo das combinações totais é o mesmo para políticas não aninhadas, exceto que precisamos considerar o nível de aninhamento de uma chave.

A combinação total é calculada da seguinte forma:

```
(2 * 2 * 2) + (2 * 2 * 2) + (2 * 2 * 2) + (2 * 2 * 2) = 32
```

`metricName` tem dois valores, `namespace` tem dois valores, `scope` é uma chave aninhada de dois níveis com um valor, `source` é uma chave aninhada de dois níveis com um valor e `type` é uma chave aninhada de dois níveis com um valor.

## Correspondência de chaves

Você pode usar o operador `exists` para estabelecer correspondência com mensagens recebidas com ou sem propriedades especificadas na política de filtro. A correspondência `exists` só funciona em nós folha. Ela não funciona em nós intermediários.

- Use `"exists": true` para estabelecer correspondência com mensagens recebidas que incluam a propriedade especificada. A chave deve ter um valor não nulo e não vazio.

Por exemplo, a propriedade de política seguinte usa o operador `exists` com um valor de `true`:

```
"store": [{"exists": true}]
```

Ela estabelece correspondência com qualquer atributo de mensagem que contenha a chave de atributo `store`, como a seguinte:

```
"store": {"Type": "String", "Value": "fans"}
"customer_interests": {"Type": "String.Array", "Value": "[\"baseball\", \"basketball\"]"}
```

Também estabelece correspondência com um dos seguintes corpos de mensagem:

```
{
  "store": "fans"
```

```
"customer_interests": ["baseball", "basketball"]
}
```

No entanto, não estabelece correspondência com nenhum atributo de mensagem sem a chave de atributo `store`, como o seguinte:

```
"customer_interests": {"Type": "String.Array", "Value": "[\"baseball\", \"basketball\"]"}
```

Nem corresponde ao seguinte corpo de mensagem:

```
{
  "customer_interests": ["baseball", "basketball"]
}
```

- Use `"exists": false` para estabelecer correspondência com mensagens recebidas que não incluam a propriedade especificada.

#### Note

`"exists": false` só estabelecerá correspondência se pelo menos um atributo estiver presente. Um conjunto vazio de atributos impossibilita que o filtro estabeleça correspondência.

Por exemplo, a propriedade de política seguinte usa o operador `exists` com um valor de `false`:

```
"store": [{"exists": false}]
```

Ela não estabelece correspondência com nenhum atributo de mensagem que contenha a chave de atributo `store`, como o seguinte:

```
"store": {"Type": "String", "Value": "fans"}
"customer_interests": {"Type": "String.Array", "Value": "[\"baseball\", \"basketball\"]"}
```

Também não estabelece correspondência com o seguinte corpo de mensagem:

```
{
```

```
"store": "fans"
"customer_interests": ["baseball", "basketball"]
}
```

No entanto, estabelece correspondência com qualquer atributo de mensagem sem a chave de atributo `store`, como o seguinte:

```
"customer_interests": {"Type": "String.Array", "Value": "[\"baseball\", \"basketball\"]"}
```

Também estabelece correspondência com o seguinte corpo de mensagem:

```
{
  "customer_interests": ["baseball", "basketball"]
}
```

## Correspondência de valores numéricos

Você pode filtrar mensagens ao estabelecer correspondência entre valores numéricos e valores de propriedades de atributos da mensagem ou do corpo da mensagem. Os valores numéricos não são inseridos entre aspas duplas na política JSON. Você pode usar as operações numéricas a seguir para filtragem.

### Note

Os prefixos são compatíveis somente com correspondência de string.

### Tópicos

- [Correspondência exata](#)
- [Correspondência anything-but](#)
- [Correspondência de intervalo de valores](#)

## Correspondência exata

Quando um valor de propriedade de política inclui a palavra-chave `numeric` e o operador `=`, estabelece correspondência com qualquer valor de propriedade de atributo ou corpo de mensagem que tenha o mesmo nome e o mesmo valor numérico.

Considere a seguinte propriedade de política:

```
"price_usd": [{"numeric": ["=", 301.5]}]
```

Ele corresponde a um dos seguintes atributos de mensagens:

```
"price_usd": {"Type": "Number", "Value": 301.5}
```

```
"price_usd": {"Type": "Number", "Value": 3.015e2}
```

Também estabelece correspondência com um dos seguintes corpos de mensagem:

```
{  
  "price_usd": 301.5  
}
```

```
{  
  "price_usd": 3.015e2  
}
```

## Correspondência anything-but

Quando o valor de uma propriedade de política inclui a palavra-chave `anything-but`, estabelece correspondência com qualquer valor de propriedade de atributo ou corpo de mensagem que não inclua nenhum dos valores de propriedade de política.

Considere a seguinte propriedade de política:

```
"price": [{"anything-but": [100, 500]}]
```

Ele corresponde a um dos seguintes atributos de mensagens:

```
"price": {"Type": "Number", "Value": 101}
```

```
"price": {"Type": "Number", "Value": 100.1}
```

Também estabelece correspondência com um dos seguintes corpos de mensagem:

```
{  
  "price": 101  
}
```

```
{  
  "price": 100.1  
}
```

Além disso, estabelece correspondência com o seguinte atributo de mensagem (pois contém um valor que não é 100 nem 500):

```
"price": {"Type": "Number.Array", "Value": "[100, 50]"}
```

Também estabelece correspondência com o seguinte corpo de mensagem (pois contém um valor que não é 100 nem 500):

```
{  
  "price": [100, 50]  
}
```

No entanto, não corresponde ao seguinte atributo de mensagem:

```
"price": {"Type": "Number", "Value": 100}
```

Nem corresponde ao seguinte corpo de mensagem:

```
{  
  "price": 100  
}
```

## Correspondência de intervalo de valores

Além do operador =, uma propriedade de política numérica pode incluir os seguintes operadores: <, <=, > e >=.

Considere a seguinte propriedade de política:

```
"price_usd": [{"numeric": ["<", 0]}]
```

Estabelece correspondência com qualquer propriedade de atributo ou corpo de mensagem com valores numéricos negativos.

Considere outro atributo de mensagem:

```
"price_usd": [{"numeric": [ ">", 0, "<=", 150 ]}]
```

Estabelece correspondência com qualquer propriedade de atributo ou corpo de mensagem com números positivos até e incluindo 150.

## Correspondência de valores de string

Você pode filtrar mensagens correspondendo valores de string a valores de atributos da mensagem ou a valores de propriedades do corpo da mensagem. Os valores de string são inseridos entre aspas duplas na política JSON. Use as operações de string a seguir para corresponder a atributos da mensagem ou ao corpo da mensagem.

### Tópicos

- [Correspondência exata](#)
- [Correspondência anything-but](#)
- [Usar um prefixo com o operador anything-but](#)
- [quals-ignore-case Combinação E](#)
- [Correspondência de endereço IP](#)
- [Correspondência de prefixo](#)
- [Correspondência de sufixo](#)

## Correspondência exata

A correspondência exata ocorre quando um valor de propriedade de política corresponde a um ou mais valores de atributos de mensagens.

Considere a seguinte propriedade de política:

```
"customer_interests": ["rugby", "tennis"]
```

Ele corresponde aos seguintes atributos de mensagem:

```
"customer_interests": {"Type": "String", "Value": "rugby"}
```

```
"customer_interests": {"Type": "String", "Value": "tennis"}
```

Também corresponde aos seguintes corpos de mensagem:

```
{  
  "customer_interests": "rugby"  
}
```

```
{  
  "customer_interests": "tennis"  
}
```

No entanto, não corresponde ao seguinte atributo de mensagem:

```
"customer_interests": {"Type": "String", "Value": "baseball"}
```

Nem corresponde ao seguinte corpo de mensagem:

```
{  
  "customer_interests": "baseball"  
}
```

## Correspondência anything-but

Quando o valor de uma propriedade de política inclui a palavra-chave `anything-but`, corresponde a qualquer valor de atributo ou corpo de mensagem que não inclua nenhum dos valores de propriedade de política. `anything-but` pode ser combinado com `"exists": false`.

Considere a seguinte propriedade de política:

```
"customer_interests": [{"anything-but": ["rugby", "tennis"]}]
```

Ele corresponde a um dos seguintes atributos de mensagens:

```
"customer_interests": {"Type": "String", "Value": "baseball"}
```

```
"customer_interests": {"Type": "String", "Value": "football"}
```

Também estabelece correspondência com um dos seguintes corpos de mensagem:

```
{  
  "customer_interests": "baseball"  
}
```

```
{  
  "customer_interests": "football"  
}
```

Além disso, estabelece correspondência com o seguinte atributo de mensagem (pois contém um valor que não é rugby nem tennis):

```
"customer_interests": {"Type": "String.Array", "Value": "[\"rugby\", \"baseball\"]"}
```

Também estabelece correspondência com o seguinte corpo de mensagem (pois contém um valor que não é rugby nem tennis):

```
{  
  "customer_interests": ["rugby", "baseball"]  
}
```

No entanto, não corresponde ao seguinte atributo de mensagem:

```
"customer_interests": {"Type": "String", "Value": "rugby"}
```

Nem corresponde ao seguinte corpo de mensagem:

```
{  
  "customer_interests": ["rugby"]  
}
```

## Usar um prefixo com o operador **anything-but**

Para correspondência de string, também é possível usar um prefixo com o operador `anything-but`. Por exemplo, a política de propriedade seguinte nega o prefixo `order-`:

```
"event": [{"anything-but": {"prefix": "order-"}}]
```

Ele corresponde a um dos seguintes atributos:

```
"event": {"Type": "String", "Value": "data-entry"}
```

```
"event": {"Type": "String", "Value": "order_number"}
```

Também estabelece correspondência com um dos seguintes corpos de mensagem:

```
{  
  "event": "data-entry"  
}
```

```
{  
  "event": "order_number"  
}
```

No entanto, não corresponde ao seguinte atributo de mensagem:

```
"event": {"Type": "String", "Value": "order-cancelled"}
```

Nem corresponde ao seguinte corpo de mensagem:

```
{  
  "event": "order-cancelled"  
}
```

## quals-ignore-case Combinação E

Quando a propriedade de uma política inclui a palavra-chave `equals-ignore-case`, ela realizará uma correspondência que não diferencia letras maiúsculas de minúsculas a qualquer valor de propriedade do corpo ou atributo da mensagem.

Considere a seguinte propriedade de política:

```
"customer_interests": [{"equals-ignore-case": "tennis"}]
```

Ele corresponde a um dos seguintes atributos de mensagens:

```
"customer_interests": {"Type": "String", "Value": "TENNIS"}
```

```
"customer_interests": {"Type": "String", "Value": "Tennis"}
```

Também estabelece correspondência com um dos seguintes corpos de mensagem:

```
{  
  "customer_interests": "TENNIS"  
}
```

```
{  
  "customer_interests": "teNnis"  
}
```

## Correspondência de endereço IP

Você pode usar o operador `cidr` para verificar se uma mensagem de entrada é originada de um endereço IP ou sub-rede específico.

Considere a seguinte propriedade de política:

```
"source_ip": [{"cidr": "10.0.0.0/24"}]
```

Ele corresponde a um dos seguintes atributos de mensagens:

```
"source_ip": {"Type": "String", "Value": "10.0.0.0"}
```

```
"source_ip": {"Type": "String", "Value": "10.0.0.255"}
```

Também estabelece correspondência com um dos seguintes corpos de mensagem:

```
{  
  "source_ip": "10.0.0.0"  
}
```

```
}
```

```
{  
  "source_ip": "10.0.0.255"  
}
```

No entanto, não corresponde ao seguinte atributo de mensagem:

```
"source_ip": {"Type": "String", "Value": "10.1.1.0"}
```

Nem corresponde ao seguinte corpo de mensagem:

```
{  
  "source_ip": "10.1.1.0"  
}
```

## Correspondência de prefixo

Quando uma propriedade de política inclui a palavra-chave `prefix`, ela corresponde a qualquer valor de atributo de mensagem ou corpo de mensagem que comece com os caracteres especificados.

Considere a seguinte propriedade de política:

```
"customer_interests": [{"prefix": "bas"}]
```

Ele corresponde a um dos seguintes atributos de mensagens:

```
"customer_interests": {"Type": "String", "Value": "baseball"}
```

```
"customer_interests": {"Type": "String", "Value": "basketball"}
```

Também estabelece correspondência com um dos seguintes corpos de mensagem:

```
{  
  "customer_interests": "baseball"  
}
```

```
{
```

```
"customer_interests": "basketball"
}
```

No entanto, não corresponde ao seguinte atributo de mensagem:

```
"customer_interests": {"Type": "String", "Value": "rugby"}
```

Nem corresponde ao seguinte corpo de mensagem:

```
{
  "customer_interests": "rugby"
}
```

## Correspondência de sufixo

Quando a propriedade de uma política inclui a palavra-chave `suffix`, ela corresponde a qualquer valor de propriedade do corpo ou atributo da mensagem que termine com os caracteres especificados.

Considere a seguinte propriedade de política:

```
"customer_interests": [{"suffix": "ball"}]
```

Ele corresponde a um dos seguintes atributos de mensagens:

```
"customer_interests": {"Type": "String", "Value": "baseball"}
```

```
"customer_interests": {"Type": "String", "Value": "basketball"}
```

Também estabelece correspondência com um dos seguintes corpos de mensagem:

```
{
  "customer_interests": "baseball"
}
```

```
{
  "customer_interests": "basketball"
}
```

No entanto, não corresponde ao seguinte atributo de mensagem:

```
"customer_interests": {"Type": "String", "Value": "rugby"}
```

Nem corresponde ao seguinte corpo de mensagem:

```
{  
  "customer_interests": "rugby"  
}
```

## Aplicar uma política de filtro de assinatura

É possível aplicar uma política de filtro a uma assinatura do Amazon SNS usando o console do Amazon SNS. Ou, para aplicar políticas de forma programática, você pode usar a API do Amazon SNS, AWS Command Line Interface o AWS CLI() ou AWS qualquer SDK compatível com o Amazon SNS. Você também pode usar AWS CloudFormation.

### Important

AWS serviços como IAM e Amazon SNS usam um modelo de computação distribuída chamado consistência eventual. As adições ou alterações a uma política de filtro de assinatura podem levar até 15 minutos para entrarem em vigor.

## AWS Management Console

1. Faça login no [console do Amazon SNS](#).
2. No painel de navegação, escolha Subscriptions (Assinaturas).
3. Selecione uma assinatura e, em seguida, escolha Edit (Editar).
4. Na página Edit (Editar) expanda a seção Subscription filter policy (Política de filtros de assinatura).
5. Escolha entre filtragem baseada em atributo ou filtragem baseada em carga útil.
6. No campo JSON editor (Editor JSON), informe o JSON body (Corpo do JSON) de sua política de filtro.
7. Escolha Save changes (Salvar alterações).

O Amazon SNS aplica a política de filtro à assinatura.

## AWS CLI

Para aplicar uma política de filtro com o AWS Command Line Interface (AWS CLI), use o [set-subscription-attributes](#) comando, conforme mostrado no exemplo a seguir. Para a opção `--attribute-name`, especifique `FilterPolicy`. Em `--attribute-value`, especifique sua política JSON.

```
$ aws sns set-subscription-attributes --subscription-arn arn:aws:sns: ... --  
attribute-name FilterPolicy --attribute-value '{"store":["example_corp"],"event":  
["order_placed"]}'
```

Para fornecer JSON válido para sua política, coloque os nomes e valores dos atributos entre aspas duplas. Também é necessário colocar o argumento completo da política entre aspas. Para evitar aspas de escape, você pode usar aspas simples para colocar a política e aspas duplas para os nomes e valores de JSON, como mostrado no exemplo.

Se você quiser alternar da filtragem de mensagens baseada em atributos (padrão) para a baseada em carga, você também pode usar o comando. [set-subscription-attributes](#) Para a opção `--attribute-name`, especifique `FilterPolicyScope`. Em `--attribute-value`, especifique `MessageBody`.

```
$ aws sns set-subscription-attributes --subscription-arn arn:aws:sns: ... --attribute-  
name FilterPolicyScope --attribute-value MessageBody
```

Para verificar se a política de filtro foi aplicada, use o comando `get-subscription-attributes`. Os atributos na saída do terminal devem mostrar a política de filtro para a chave `FilterPolicy`, conforme mostrado no exemplo a seguir:

```
$ aws sns get-subscription-attributes --subscription-arn arn:aws:sns: ...  
{  
  "Attributes": {  
    "Endpoint": "endpoint . . .",  
    "Protocol": "https",  
    "RawMessageDelivery": "false",  
    "EffectiveDeliveryPolicy": "delivery policy . . .",  
    "ConfirmationWasAuthenticated": "true",  
    "FilterPolicy": "{\"store\": [\"example_corp\"], \"event\": [\"order_placed  
\"]}",  
    "FilterPolicyScope": "MessageAttributes",  
    "Owner": "111122223333",
```

```
    "SubscriptionArn": "arn:aws:sns: . . .",
    "TopicArn": "arn:aws:sns: . . ."
  }
}
```

## AWS SDKs

Os exemplos de código a seguir mostram como usar `setSubscriptionAttributes`.

### Important

Se você estiver usando o exemplo do SDK para Java 2.x, a classe `SNSMessageFilterPolicy` não estará disponível por padrão. Para obter instruções sobre como instalar essa classe, consulte o [exemplo](#) do GitHub site.

## CLI

### AWS CLI

Para definir atributos de assinatura

O exemplo `set-subscription-attributes` a seguir define o atributo `RawMessageDelivery` para uma assinatura do SQS.

```
aws sns set-subscription-attributes \
  --subscription-arn arn:aws:sns:us-
east-1:123456789012:mytopic:f248de18-2cf6-578c-8592-b6f1eaa877dc \
  --attribute-name RawMessageDelivery \
  --attribute-value true
```

Este comando não produz saída.

O exemplo `set-subscription-attributes` a seguir define um atributo `FilterPolicy` para uma assinatura do SQS.

```
aws sns set-subscription-attributes \
  --subscription-arn arn:aws:sns:us-
east-1:123456789012:mytopic:f248de18-2cf6-578c-8592-b6f1eaa877dc \
  --attribute-name FilterPolicy \
  --attribute-value "{ \"anyMandatoryKey\": [\"any\", \"of\", \"these\"] }"
```

Este comando não produz saída.

O exemplo `set-subscription-attributes` a seguir remove o atributo `FilterPolicy` de uma assinatura do SQS.

```
aws sns set-subscription-attributes \  
  --subscription-arn arn:aws:sns:us-  
east-1:123456789012:mytopic:f248de18-2cf6-578c-8592-b6f1eaa877dc \  
  --attribute-name FilterPolicy \  
  --attribute-value "{}"
```

Este comando não produz saída.

- Para obter detalhes da API, consulte [SetSubscriptionAttributes](#) na Referência de AWS CLI Comandos.

## Java

### SDK para Java 2.x

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```
import software.amazon.awssdk.regions.Region;  
import software.amazon.awssdk.services.sns.SnsClient;  
import software.amazon.awssdk.services.sns.model.SnsException;  
import java.util.ArrayList;  
  
/**  
 * Before running this Java V2 code example, set up your development  
 * environment, including your credentials.  
 *  
 * For more information, see the following documentation topic:  
 *  
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html  
 */  
public class UseMessageFilterPolicy {
```

```
public static void main(String[] args) {
    final String usage = ""

        Usage:    <subscriptionArn>

        Where:
            subscriptionArn - The ARN of a subscription.

        """;

    if (args.length != 1) {
        System.out.println(usage);
        System.exit(1);
    }

    String subscriptionArn = args[0];
    SnsClient snsClient = SnsClient.builder()
        .region(Region.US_EAST_1)
        .build();

    usePolicy(snsClient, subscriptionArn);
    snsClient.close();
}

public static void usePolicy(SnsClient snsClient, String subscriptionArn) {
    try {
        SNSMessageFilterPolicy fp = new SNSMessageFilterPolicy();
        // Add a filter policy attribute with a single value
        fp.addAttribute("store", "example_corp");
        fp.addAttribute("event", "order_placed");

        // Add a prefix attribute
        fp.addAttributePrefix("customer_interests", "bas");

        // Add an anything-but attribute
        fp.addAttributeAnythingBut("customer_interests", "baseball");

        // Add a filter policy attribute with a list of values
        ArrayList<String> attributeValues = new ArrayList<>();
        attributeValues.add("rugby");
        attributeValues.add("soccer");
        attributeValues.add("hockey");
        fp.addAttribute("customer_interests", attributeValues);
    }
}
```

```
// Add a numeric attribute
fp.addAttribute("price_usd", "=", 0);

// Add a numeric attribute with a range
fp.addAttributeRange("price_usd", ">", 0, "<=", 100);

// Apply the filter policy attributes to an Amazon SNS subscription
fp.apply(snsClient, subscriptionArn);

} catch (SnsException e) {
    System.err.println(e.awsErrorDetails().errorMessage());
    System.exit(1);
}
}
```

- Para obter detalhes da API, consulte [SetSubscriptionAttributes](#) a Referência AWS SDK for Java 2.x da API.

## Python

### SDK para Python (Boto3)

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```
class SnsWrapper:
    """Encapsulates Amazon SNS topic and subscription functions."""

    def __init__(self, sns_resource):
        """
        :param sns_resource: A Boto3 Amazon SNS resource.
        """
        self.sns_resource = sns_resource

    @staticmethod
```

```
def add_subscription_filter(subscription, attributes):
    """
    Adds a filter policy to a subscription. A filter policy is a key and a
    list of values that are allowed. When a message is published, it must
    have an
    attribute that passes the filter or it will not be sent to the
    subscription.

    :param subscription: The subscription the filter policy is attached to.
    :param attributes: A dictionary of key-value pairs that define the
    filter.
    """
    try:
        att_policy = {key: [value] for key, value in attributes.items()}
        subscription.set_attributes(
            AttributeName="FilterPolicy",
            AttributeValue=json.dumps(att_policy)
        )
        logger.info("Added filter to subscription %s.", subscription.arn)
    except ClientError:
        logger.exception(
            "Couldn't add filter to subscription %s.", subscription.arn
        )
        raise
```

- Para obter detalhes da API, consulte a [SetSubscriptionAttributes](#) Referência da API AWS SDK for Python (Boto3).

## API do Amazon SNS

Para aplicar um política de filtro com a API do Amazon SNS, faça uma solicitação para a ação [SetSubscriptionAttributes](#). Defina o parâmetro `AttributeName` como `FilterPolicy` e defina o parâmetro `AttributeValue` para a política de filtro JSON.

Se você quiser mudar da filtragem baseada em atributos (padrão) para a filtragem baseada em carga útil, também poderá usar a ação [SetSubscriptionAttributes](#). Defina o parâmetro `AttributeName` como `FilterPolicyScope` e defina o parâmetro `AttributeValue` como `MessageBody`.

## AWS CloudFormation

Para aplicar uma política de filtro usando AWS CloudFormation, use um modelo JSON ou YAML para criar uma AWS CloudFormation pilha. Para obter mais informações, consulte a [FilterPolicypropriedade](#) do `AWS::SNS::Subscription` recurso no Guia do AWS CloudFormation usuário e no [AWS CloudFormation modelo de exemplo](#).

1. Faça login no [console do AWS CloudFormation](#).
2. Escolha Create Stack (Criar pilha).
3. Na página Select Template (Selecionar modelo), escolha Upload a template to Amazon S3 (Carregar um modelo no Amazon S3) e, em seguida, o arquivo e Next (Avançar).
4. Na página Specify Details (Especificar detalhes), faça o seguinte:
  - a. Em Stack Name (Nome da pilha), digite `MyFilterPolicyStack`.
  - b. Para `myHttpEndpoint`, digite o endpoint HTTP a ser inscrito em seu tópico.

### Tip

Se você não tiver um endpoint HTTP, crie um.

5. Na página Options (Opções), escolha Next (Avançar).
6. Na página Review (Revisar), escolha Create (Criar).

## Remover uma política de filtro de assinatura

Para interromper a filtragem de mensagens que são enviadas para uma assinatura, remova a política de filtro da assinatura substituindo-a por um corpo JSON vazio. Após a remoção da política, a assinatura aceitará todas as mensagens que forem publicadas nela.

## AWS Management Console

1. Faça login no [console do Amazon SNS](#).
2. No painel de navegação, escolha Subscriptions (Assinaturas).
3. Selecione uma assinatura e, em seguida, escolha Edit (Editar).
4. Na página Edit **EXAMPLE1-23bc-4567-d890-ef12g3hij456** (Editar EXAMPLE1-23bc-4567-d890-ef12g3hij456), expanda a seção Subscription filter policy (Política de filtros de assinatura).

5. No campo JSON editor (Editor JSON), forneça um corpo do JSON vazio para seu filtro de política: {}.
6. Escolha Save changes (Salvar alterações).

O Amazon SNS aplica a política de filtro à assinatura.

## AWS CLI

Para remover uma política de filtro com a AWS CLI, use o comando [set-subscription-attributes](#) e forneça um corpo JSON vazio para o argumento `--attribute-value`:

```
$ aws sns set-subscription-attributes --subscription-arn arn:aws:sns:... --attribute-name FilterPolicy --attribute-value "{}"
```

## API do Amazon SNS

Para remover uma política de filtro com a API do Amazon SNS, faça uma solicitação para a ação [SetSubscriptionAttributes](#). Defina o parâmetro `AttributeName` como `FilterPolicy` e forneça um corpo JSON vazio para o parâmetro `AttributeValue`.

# Proteção de dados de mensagens

## Tópicos

- [O que é proteção de dados de mensagens?](#)
- [Por que devo usar a proteção de dados de mensagens?](#)
- [Noções básicas sobre políticas de proteção de dados](#)
- [Identificadores de dados](#)

## O que é proteção de dados de mensagens?

A proteção de dados de mensagens garante a segurança dos dados publicados nos tópicos do Amazon SNS usando [políticas de proteção de dados](#) para auditar, mascarar, eliminar ou bloquear as informações sigilosas transmitidas entre aplicações ou serviços da AWS.

A proteção de dados de mensagens verifica dados em movimento em busca de informações de identificação pessoal (PII) e informações de saúde protegidas (PHI) usando identificadores de dados. É possível optar por usar identificadores de dados [predefinidos](#) (ou gerenciados pelo Amazon SNS) (por exemplo, nomes, endereços, números de cartão de crédito e códigos de medicamentos prescritos) ou criar seus próprios identificadores de dados [personalizados](#) especificamente para o caso de uso de negócios. Usando as informações verificadas, a proteção de dados de mensagens fornece logs de auditoria detalhados e permite que você tome medidas para proteger esses dados.

A proteção de dados de mensagens comporta as seguintes ações para ajudar a proteger as informações sigilosas do cliente:

- [Audit](#) (Auditoria): audite até 99% dos dados publicados em um tópico do Amazon SNS. Em seguida, você pode optar por enviar as descobertas para a [Amazon CloudWatch](#), [Amazon S3](#) ou [Amazon Data Firehose](#).
- [Desidentificar](#): mascara ou edita dados confidenciais sem interromper a publicação ou a entrega de mensagens.
- [Deny](#) (Negar): bloqueie a transmissão de dados entre aplicações e recursos da AWS se houver dados sigilosos na carga útil.

**Note**

O Amazon SNS é compatível com a proteção de dados de mensagens somente para tópicos padrão do Amazon SNS.

## Por que devo usar a proteção de dados de mensagens?

Ao introduzir a proteção de dados de mensagens em seus programas de governança, gerenciamento de riscos e conformidade, você pode implementar políticas de proteção de dados que ajudam a identificar e evitar o vazamento de dados. Isso oferece às suas equipes ferramentas que podem ajudar a reduzir riscos financeiros, jurídicos e regulatórios por meio do cumprimento de regulamentos de privacidade como HIPAA, RGPD, PCI e FedRAMP. Também libera seus desenvolvedores da sobrecarga operacional associada à criação e ao gerenciamento de suas próprias ferramentas para proteger dados sigilosos.

Por exemplo, você pode usar a proteção de dados de mensagens para criar uma política de auditoria destinada a determinar se algum de seus sistemas está acidentalmente enviando ou recebendo dados sigilosos. Se os resultados da auditoria mostrarem que os sistemas estão enviando informações de cartão de crédito a sistemas que não precisam delas, você pode usar uma política de bloqueio para impedir a entrega dos dados.

**Note**

O Amazon SNS é compatível com a proteção de dados de mensagens somente para tópicos padrão do Amazon SNS.

## Noções básicas sobre políticas de proteção de dados

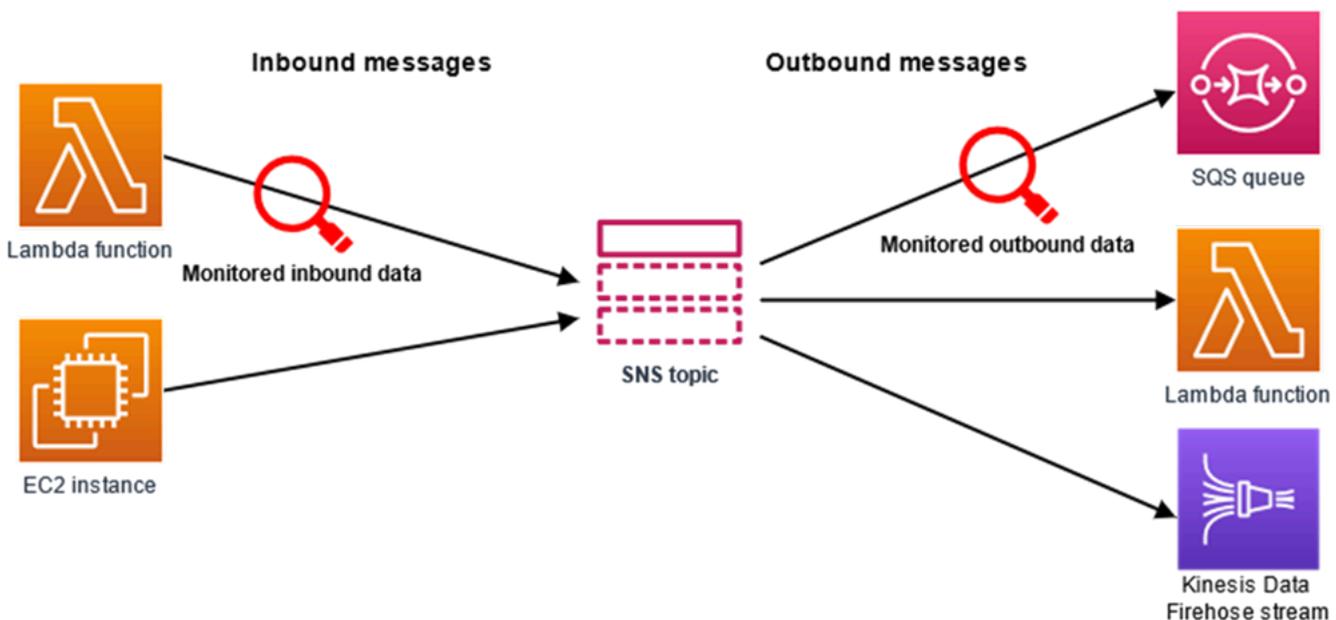
### Tópicos

- [O que são políticas de proteção de dados?](#)
- [Como a política de proteção de dados é estruturada?](#)
- [Como determino as entidades principais do IAM para minha política de proteção de dados?](#)
- [Operações de política de proteção de dados](#)
- [Exemplos de política de proteção de dados](#)

- [Criar políticas de proteção de dados](#)
- [Excluir políticas de proteção de dados no Amazon SNS](#)

## O que são políticas de proteção de dados?

O Amazon SNS usa políticas de proteção de dados para selecionar os dados sigilosos que você deseja verificar e as ações que quer realizar para evitar que esses dados sejam trocados por seus tópicos do Amazon SNS. Para selecionar os dados sigilosos de interesse, você deve usar [identificadores de dados](#). Depois, a proteção de dados de mensagens do Amazon SNS detecta os dados sigilosos usando machine learning e correspondência de padrões. Para agir de acordo com os identificadores de dados encontrados, você pode definir uma operação de auditoria, desidentificação ou negação. Essas operações permitem que você registre os dados confidenciais encontrados (ou não encontrados), mascare ou elimine os dados confidenciais ou negue a entrega de mensagens.

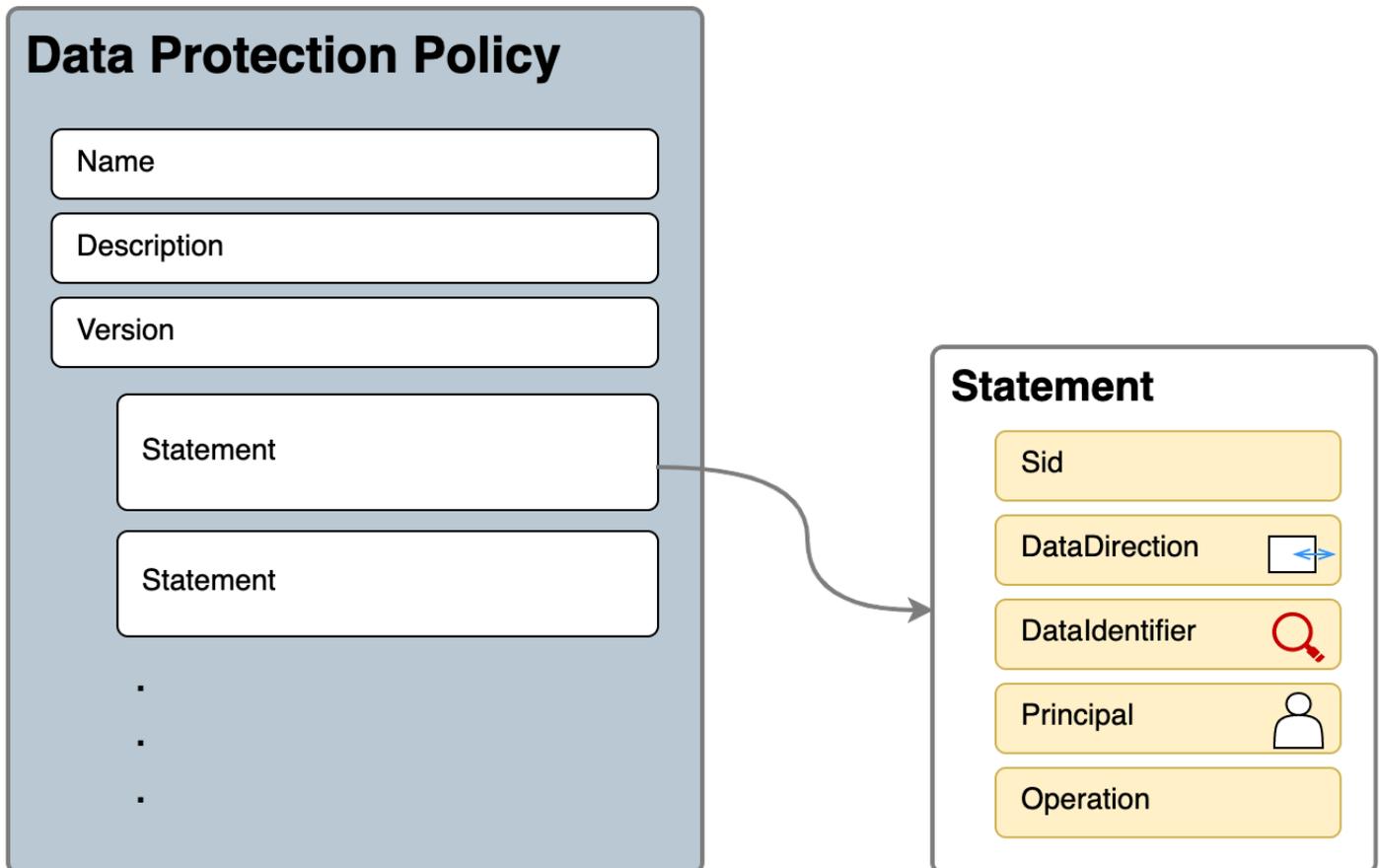


## Como a política de proteção de dados é estruturada?

Como mostrado na figura abaixo, um documento de política de proteção de dados inclui os seguintes elementos:

- Informações opcionais da política na parte superior do documento
- Uma ou mais instruções individuais

Cada instrução inclui informações sobre uma única permissão.



Só é possível definir uma política de proteção de dados por tópico do Amazon SNS. A política de proteção de dados pode ter uma ou mais instruções de negação ou desidentificação, mas somente uma instrução de auditoria.

## Propriedades JSON para a política de proteção de dados

Uma política de proteção de dados requer as seguintes informações básicas de política para identificação:

- Nome: o nome da política.
- Descrição (Opcional): a descrição da política.
- Versão: a versão do idioma das políticas. A versão atual é 2021-06-01.
- Declaração: uma lista de declarações que especificam as ações da política de proteção de dados.

```
{
  "Name": "basicPII-protection",
  "Description": "Protect basic types of sensitive data",
```

```
"Version": "2021-06-01",
"Statement": [
    ...
]
}
```

## Propriedades JSON para uma declaração de política

Uma declaração de política define o contexto de detecção para a operação de proteção de dados.

- **Sid** (Opcional): o identificador da declaração.
- **DataDirection**: entrada (para solicitações de API de publicação) ou saída (para entregas de notificações) com relação ao tópico do Amazon SNS.
- **DataIdentifier**: os dados sigilosos que o tópico do Amazon SNS deve examinar. Por exemplo, nome, endereço ou número de telefone.
- **Entidade principal**: a entidade principal do IAM que publicou no tópico ou a entidade principal do IAM que assinou o tópico.
- **Operação**: a ação subsequente, seja **Audit** (Auditor), **De-identify** (Desidentificar) (mascarar ou eliminar) ou **Deny** (Negar), que o tópico do Amazon SNS realiza quando encontra dados confidenciais.

```
{
  "Sid": "basicPII-inbound-protection",
  "DataDirection": "Inbound",
  "Principal": ["*"],
  "DataIdentifier": [
    "arn:aws:dataprotection::aws:data-identifier/Name",
    "arn:aws:dataprotection::aws:data-identifier/PhoneNumber-US"
  ],
  "Operation": {
    ...
  }
}
```

## Propriedades JSON para uma operação de declaração de política

Uma declaração de política define uma das operações de proteção de dados a seguir.

- [Audit](#) (Auditoria): emite métricas e localiza logs sem interromper a publicação ou a entrega de mensagens.
- [Desidentificar](#): mascara ou elimina dados confidenciais sem interromper a publicação de mensagens.
- [Deny](#) (Negação): bloqueia a solicitação de publicação do Amazon SNS ou não entrega a mensagem.

## Como determino as entidades principais do IAM para minha política de proteção de dados?

A proteção de dados de mensagens usa duas entidades principais do IAM que interagem com o Amazon SNS.

1. Entidade principal da API de publicação (entrada): a entidade principal autenticada do IAM que chama a API Publish do Amazon SNS.
2. Entidades principais de assinatura (saída): a entidade principal autenticada do IAM que chamou a API Subscribe durante a criação da assinatura.

A `SubscriptionPrincipal` é uma propriedade de assinatura do Amazon SNS disponível ao público que pode ser recuperada na API `GetSubscriptionAttributes`.

```
{
  "Attributes": {
    "SubscriptionPrincipal": "arn:aws:iam::123456789012:user/NoNameAccess",
    "Owner": "123412341234",
    "RawMessageDelivery": "true",
    "TopicArn": "arn:aws:sns:us-east-1:123412341234:PII-data-topic",
    "Endpoint": "arn:aws:sqs:us-east-1:123456789012:NoNameAccess",
    "Protocol": "sqs",
    "PendingConfirmation": "false",
    "ConfirmationWasAuthenticated": "true",
    "SubscriptionArn": "arn:aws:sns:us-east-1:123412341234:PII-data-
topic:5d8634ef-67ef-49eb-a824-4042b28d6f55"
  }
}
```

## Operações de política de proteção de dados

Veja a seguir as políticas de proteção de dados que você pode usar para auditar e negar dados sigilosos. Para ver um tutorial completo que inclui uma aplicação de exemplo, consulte a publicação do blog [Introducing message data protection for Amazon SNS](#) (“Apresentar a proteção de dados de mensagens para o Amazon SNS”).

### Tópicos

- [Operação de auditoria](#)
- [Operação de desidentificação](#)
- [Operação de negação](#)

### Operação de auditoria

A operação Audit (Auditoria) examina uma amostra das mensagens de entrada do tópico e registra as descobertas de dados sigilosos em um destino da AWS. A taxa de amostragem pode ser um número inteiro entre 0 e 99. Essa operação requer um dos seguintes tipos de destino de registro em log:

1. FindingsDestination— O destino do registro quando o tópico do Amazon SNS encontra dados confidenciais na carga útil.
2. NoFindingsDestination— O destino do registro quando o tópico do Amazon SNS não encontra dados confidenciais na carga útil.

É possível usar os seguintes Serviços da AWS em cada um destes tipos de destino de log:

- Amazon CloudWatch Logs (opcional) — Eles LogGroup devem estar na região do tópico e o nome deve começar com /aws/vendedlogs/.
- Amazon Data Firehose (opcional) — DeliveryStream Deve estar na região do tópico e ter o Direct PUT como fonte do fluxo de entrega. Para obter detalhes adicionais, consulte [Origem, destino e nome](#) no Guia do desenvolvedor do Amazon Data Firehose.
- Amazon S3 (Opcional): um nome de bucket do Amazon S3. [Ações extras são necessárias para usar o bucket do Amazon S3 com a criptografia SSE-KMS ativada.](#)

```
{  
  "Operation": {
```

```

"Audit": {
  "SampleRate": "99",
  "FindingsDestination": {
    "CloudWatchLogs": {
      "LogGroup": "/aws/vendedlogs/log-group-name"
    },
    "Firehose": {
      "DeliveryStream": "delivery-stream-name"
    },
    "S3": {
      "Bucket": "bucket-name"
    }
  },
  "NoFindingsDestination": {
    "CloudWatchLogs": {
      "LogGroup": "/aws/vendedlogs/log-group-name"
    },
    "Firehose": {
      "DeliveryStream": "delivery-stream-name"
    },
    "S3": {
      "Bucket": "bucket-name"
    }
  }
}
}
}
}

```

### Permissões necessárias ao especificar destinos de log

Ao especificar destinos de registro em log na política de proteção de dados, você deve adicionar as seguintes permissões à política de identidade do IAM da entidade principal do IAM que está chamando o Amazon SNS: `API PutDataProtectionPolicy` ou `CreateTopic` com o parâmetro `--data-protection-policy`.

Destino de auditoria	Permissão do IAM
Padrão	logs:CreateLogDelivery
	logs:GetLogDelivery
	logs:UpdateLogDelivery

Destino de auditoria	Permissão do IAM
	logs:DeleteLogDelivery logs:ListLogDeliveries
CloudWatchLogs	logs:PutResourcePolicy logs:DescribeResourcePolicies logs:DescribeLogGroups
Firehose	iam:CreateServiceLinkedRole firehose:TagDeliveryStream
S3	s3:PutBucketPolicy s3:GetBucketPolicy <a href="#">Ações extras são necessárias para usar o bucket do Amazon S3 com a criptografia SSE-KMS ativada.</a>

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "logs:CreateLogDelivery",
        "logs:GetLogDelivery",
        "logs:UpdateLogDelivery",
        "logs:DeleteLogDelivery",
        "logs:ListLogDeliveries"
      ],
      "Resource": [
        "*"
      ]
    },
    {
      "Effect": "Allow",
```

```
"Action": [
  "logs:PutResourcePolicy",
  "logs:DescribeResourcePolicies",
  "logs:DescribeLogGroups"
],
"Resource": [
  "arn:aws:logs:region:account-id:SampleLogGroupName:*:*"
]
},
{
  "Effect": "Allow",
  "Action": [
    "iam:CreateServiceLinkedRole",
    "firehose:TagDeliveryStream"
  ],
  "Resource": "*"
},
{
  "Effect": "Allow",
  "Action": [
    "s3:PutBucketPolicy",
    "s3:GetBucketPolicy"
  ],
  "Resource": [
    "arn:aws:s3:::bucket-name"
  ]
}
]
```

## Política de chaves obrigatórias para uso com SSE-KMS

Se você usa um bucket do Amazon S3 como destino de log, é possível proteger os dados no bucket habilitando a criptografia do lado do servidor com chaves gerenciadas pelo Amazon S3 (SSE-S3) ou a criptografia do lado do servidor com chaves do AWS KMS keys (SSE-KMS). Para obter mais informações, consulte [Proteger dados usando criptografia do lado do servidor](#) no Manual do usuário do Amazon S3.

Se você escolher SSE-S3, nenhuma configuração adicional será necessária. O Amazon S3 lida com a chave de criptografia.

Se você escolher o SSE-KMS, deverá usar uma chave gerenciada pelo cliente. Você deve atualizar a política de chaves para a chave gerenciada pelo cliente para que a conta de entrega de logs possa

gravar no bucket do S3. Para obter mais informações sobre a política de chaves necessária para uso com o SSE-KMS, consulte a [criptografia do lado do servidor do bucket Amazon S3](#) no Guia do usuário do Amazon Logs. CloudWatch

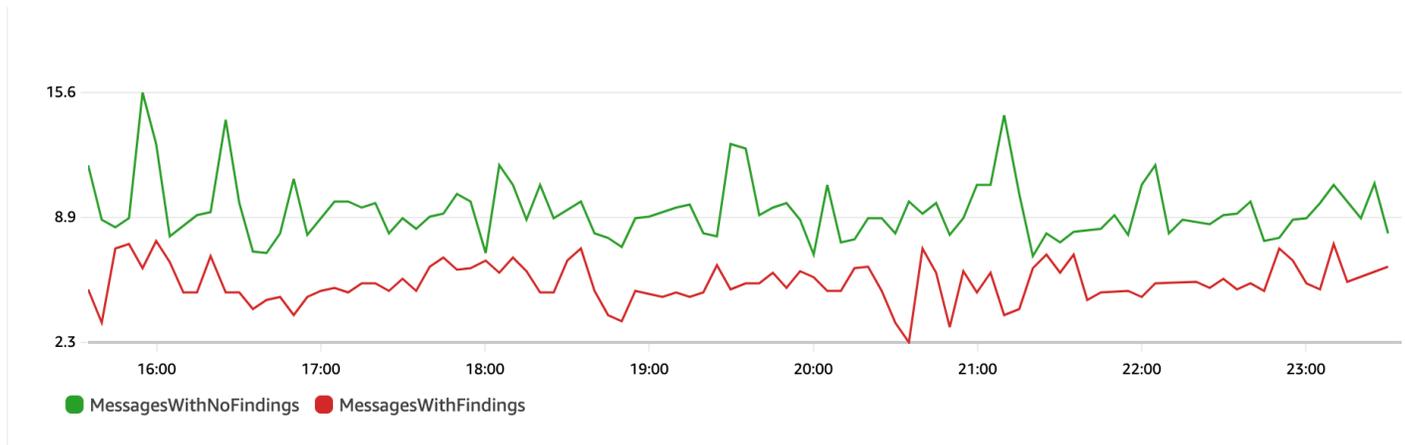
### Exemplo de log de destino de auditoria

No exemplo a seguir, `callerPrincipal` é usado para identificar a origem do conteúdo sigiloso e `messageID` como referência para comparar com a resposta da API Publish.

```
{
  "messageId": "34d9b400-c6dd-5444-820d-fbeb0f1f54cf",
  "auditTimestamp": "2022-05-12T2:10:44Z",
  "callerPrincipal": "arn:aws:iam::123412341234:role/Publisher",
  "resourceArn": "arn:aws:sns:us-east-1:123412341234:PII-data-topic",
  "dataIdentifiers": [
    {
      "name": "Name",
      "count": 1,
      "detections": [
        {
          "start": 1,
          "end": 2
        }
      ]
    },
    {
      "name": "PhoneNumber",
      "count": 2,
      "detections": [
        {
          "start": 3,
          "end": 4
        },
        {
          "start": 5,
          "end": 6
        }
      ]
    }
  ]
}
```

## Métricas de operação de auditoria

Quando uma operação de auditoria especifica a `FindingsDestination` ou a `NoFindingsDestination` propriedade, os proprietários do tópico também recebem `CloudWatch MessagesWithFindings` `MessagesWithNoFindings` métricas.



## Operação de desidentificação

A operação Remoção de identificação mascara ou elimina dados sigilosos das mensagens publicadas ou entregues. Essa operação está disponível para mensagens de entrada e saída e requer um dos seguintes tipos de configuração:

- **MaskConfig**— Máscara usando um caractere compatível da tabela a seguir. Por exemplo, `ssn: 123-45-6789` se torna `ssn: #####`.

```
{
  "Operation": {
    "Deidentify": {
      "MaskConfig": {
        "MaskWithCharacter": "#"
      }
    }
  }
}
```

Caractere de máscara compatível	Nome
*	Asterisco
A–Z, a–z e 0–9	Alfanumérico

Caractere de máscara compatível	Nome
	Espaço
!	Ponto de exclamação
\$	Sinal de dólar
%	Sinal de porcentagem
&	E comercial
()	Parênteses
+	Sinal de adição
,	Vírgula
-	Hífen
.	Período
^	Barra, barra invertida
#	Sinal numérico
:	Dois pontos
;	Ponto e vírgula
=, <>	Igual, menor que ou maior que
@	Arroba
[]	Colchetes
^	Circunflexo
—	Sublinhado
`	Acento grave

Caractere de máscara compatível	Nome
	Barra vertical
~	Til

- **RedactConfig**— Redija removendo totalmente os dados. Por exemplo, ssn: 123-45-6789 se torna ssn: .

```
{
  "Operation": {
    "Deidentify": {
      "RedactConfig": {}
    }
  }
}
```

Em uma mensagem de entrada, os dados confidenciais são desidentificados após a operação de auditoria, e o chamador da API `SNS:Publish` recebe o erro de parâmetro inválido a seguir quando toda a mensagem é confidencial.

Error code: `AuthorizationError` ...

## Operação de negação

A operação `Deny` (Negação) interromperá a solicitação da API `Publish` ou a entrega da mensagem se a mensagem contiver dados sigilosos. O objeto da operação `Deny` (Negação) está em branco, pois não requer configuração adicional.

```
"Operation": {
  "Deny": {}
}
```

Em uma mensagem de entrada, o chamador da API `SNS:Publish` recebe um erro de autorização.

Error code: `AuthorizationError` ...

Em uma mensagem de saída, o tópico do Amazon SNS não entrega a mensagem para a assinatura. Para monitorar entregas não autorizadas, habilite o [registro em log do status de entrega](#) do tópico. Veja a seguir um exemplo de log de status de entrega:

```
{
  "notification": {
    "messageMD5Sum": "29638742fffb68b32cf56f42a79bcf16b",
    "messageId": "34d9b400-c6dd-5444-820d-fbeb0f1f54cf",
    "topicArn": "arn:aws:sns:us-east-1:123412341234:PII-data-topic",
    "timestamp": "2022-05-12T2:12:44Z"
  },
  "delivery": {
    "deliveryId": "98236591c-56aa-51ee-a5ed-0c7d43493170",
    "destination": "arn:aws:sqs:us-east-1:123456789012:NoNameAccess",
    "providerResponse": "The topic's data protection policy prohibits this message
from being delivered to <subscription-arn>",
    "dwellTimeMs":20,
    "attempts":1,
    "statusCode": 403
  },
  "status": "FAILURE"
}
```

## Exemplos de política de proteção de dados

Veja a seguir as políticas de proteção de dados que você pode usar para auditar e negar dados sigilosos. Para ver um tutorial completo que inclui uma aplicação de exemplo, consulte a publicação do blog [Introducing message data protection for Amazon SNS](#) (“Apresentar a proteção de dados de mensagens para o Amazon SNS”).

### Tópicos

- [Exemplo de política para auditoria](#)
- [Exemplo de política com instrução de máscara de desidentificação de entrada](#)
- [Exemplo de política com instrução de eliminação da desidentificação de entrada](#)
- [Exemplo de política com instrução de máscara da desidentificação de saída](#)
- [Exemplo de política com instrução de edição da desidentificação de saída](#)
- [Exemplo de política com instrução de negação de entrada](#)
- [Exemplo de política com instrução de negação de saída](#)

## Exemplo de política para auditoria

As políticas de auditoria permitem auditar até 99% das mensagens recebidas e enviar descobertas para a [Amazon CloudWatch](#), [Amazon Data Firehose](#) e Amazon [S3](#).

Por exemplo, você pode criar uma política de auditoria para avaliar se algum de seus sistemas está acidentalmente enviando ou recebendo dados sigilosos. Se os resultados da auditoria mostrarem que os sistemas estão enviando informações de cartão de crédito a sistemas que não precisam delas, você pode implementar uma política de proteção de dados para bloquear a entrega dos dados.

O exemplo a seguir audita 99% das mensagens que fluem pelo tópico procurando números de cartão de crédito e enviando as descobertas para CloudWatch Logs, Firehose e Amazon S3.

Política de proteção de dados:

```
{
  "Name": "__example_data_protection_policy",
  "Description": "Example data protection policy",
  "Version": "2021-06-01",
  "Statement": [
    {
      "DataDirection": "Inbound",
      "Principal": ["*"],
      "DataIdentifier": [
        "arn:aws:dataprotection::aws:data-identifier/CreditCardNumber"
      ],
      "Operation": {
        "Audit": {
          "SampleRate": "99",
          "FindingsDestination": {
            "CloudWatchLogs": {
              "LogGroup": "<example log name>"
            },
            "Firehose": {
              "DeliveryStream": "<example stream name>"
            },
            "S3": {
              "Bucket": "<example bucket name>"
            }
          }
        }
      }
    }
  ]
}
```

```

    }
  }
]
}

```

Exemplo de formato de resultados de auditoria:

```

{
  "messageId": "...",
  "callerPrincipal": "arn:aws:sts::123456789012:assumed-role/ExampleRole",
  "resourceArn": "arn:aws:sns:us-east-1:123456789012:ExampleArn",
  "dataIdentifiers": [
    {
      "name": "CreditCardNumber",
      "count": 1,
      "detections": [
        { "start": 1, "end": 2 }
      ]
    }
  ],
  "timestamp": "2021-04-20T00:33:40.241Z"
}

```

Exemplo de política com instrução de máscara de desidentificação de entrada

O exemplo a seguir impede que um usuário publique uma mensagem em um tópico com `CreditCardNumber` ao mascarar os dados confidenciais do conteúdo da mensagem.

```

{
  "Name": "__example_data_protection_policy",
  "Description": "Example data protection policy",
  "Version": "2021-06-01",
  "Statement": [
    {
      "DataDirection": "Inbound",
      "Principal": [
        "arn:aws:iam::123456789012:user/ExampleUser"
      ],
      "DataIdentifier": [
        "arn:aws:dataprotection::aws:data-identifier/CreditCardNumber"
      ],
    }
  ],
}

```

```

    "Operation": {
      "Deidentify": {
        "MaskConfig": {
          "MaskWithCharacter": "#"
        }
      }
    }
  }
]
}

```

Exemplo de resultados da máscara de desidentificação de entrada:

```

// original message
My credit card number is 4539894458086459

// delivered message
My credit card number is #####

```

## Exemplo de política com instrução de eliminação da desidentificação de entrada

O exemplo a seguir impede que um usuário publique uma mensagem em um tópico com `CreditCardNumber` ao eliminar os dados confidenciais do conteúdo da mensagem.

```

{
  "Name": "__example_data_protection_policy",
  "Description": "Example data protection policy",
  "Version": "2021-06-01",
  "Statement": [
    {
      "DataDirection": "Inbound",
      "Principal": [
        "arn:aws:iam::123456789012:user/ExampleUser"
      ],
      "DataIdentifier": [
        "arn:aws:dataprotection::aws:data-identifier/CreditCardNumber"
      ],
      "Operation": {
        "Deidentify": {
          "RedactConfig": {}
        }
      }
    }
  ]
}

```

```

    }
  ]
}

```

Exemplo de resultados de eliminação de desidentificação de entrada:

```

// original message
My credit card number is 4539894458086459

// delivered message
My credit card number is

```

Exemplo de política com instrução de máscara da desidentificação de saída

O exemplo a seguir impede que um usuário receba uma mensagem com `CreditCardNumber` ao mascarar os dados sigilosos do respectivo conteúdo.

```

{
  "Name": "__example_data_protection_policy",
  "Description": "Example data protection policy",
  "Version": "2021-06-01",
  "Statement": [
    {
      "DataDirection": "Outbound",
      "Principal": [
        "arn:aws:iam::123456789012:user/ExampleUser"
      ],
      "DataIdentifier": [
        "arn:aws:dataprotection::aws:data-identifier/CreditCardNumber"
      ],
      "Operation": {
        "Deidentify": {
          "MaskConfig": {
            "MaskWithCharacter": "-"
          }
        }
      }
    }
  ]
}

```

Exemplo de resultados da máscara de desidentificação de saída:

```
// original message
My credit card number is 4539894458086459

// delivered message
My credit card number is -----
```

## Exemplo de política com instrução de edição da desidentificação de saída

O exemplo a seguir impede que um usuário receba uma mensagem com `CreditCardNumber` ao eliminar os dados sigilosos do conteúdo da mensagem.

```
{
  "Name": "__example_data_protection_policy",
  "Description": "Example data protection policy",
  "Version": "2021-06-01",
  "Statement": [
    {
      "DataDirection": "Outbound",
      "Principal": [
        "arn:aws:iam::123456789012:user/ExampleUser"
      ],
      "DataIdentifier": [
        "arn:aws:dataprotection::aws:data-identifier/CreditCardNumber"
      ],
      "Operation": {
        "Deidentify": {
          "RedactConfig": {}
        }
      }
    }
  ]
}
```

Exemplo de resultados de eliminação de desidentificação de saída:

```
// original message
My credit card number is 4539894458086459

// delivered message
My credit card number is
```

## Exemplo de política com instrução de negação de entrada

O exemplo a seguir impede que um usuário publique uma mensagem em um tópico com `CreditCardNumber` no conteúdo da mensagem. As cargas úteis negadas na resposta da API têm o código de status "403 AuthorizationError".

```
{
  "Name": "__example_data_protection_policy",
  "Description": "Example data protection policy",
  "Version": "2021-06-01",
  "Statement": [
    {
      "DataDirection": "Inbound",
      "Principal": [
        "arn:aws:iam::123456789012:user/ExampleUser"
      ],
      "DataIdentifier": [
        "arn:aws:dataprotection::aws:data-identifier/CreditCardNumber"
      ],
      "Operation": {
        "Deny": {}
      }
    }
  ]
}
```

## Exemplo de política com instrução de negação de saída

O exemplo a seguir impede que uma conta da AWS receba mensagens que contenham `CreditCardNumber`.

```
{
  "Name": "__example_data_protection_policy",
  "Description": "Example data protection policy",
  "Version": "2021-06-01",
  "Statement": [
    {
      "DataDirection": "Outbound",
      "Principal": [
        "arn:aws:iam::123456789012:user/ExampleUser"
      ],
      "DataIdentifier": [
```

```

    "arn:aws:dataprotection::aws:data-identifier/CreditCardNumber"
  ],
  "Operation": {
    "Deny": {}
  }
}
]
}

```

## Exemplo de resultados de negação de saída, registrados na Amazon: CloudWatch

```

{
  "notification": {
    "messageMD5Sum": "2e8f58ff2eed723b56b15493fbfb5a5",
    "messageId": "8747a956-ebf1-59da-b291-f2c2e4b87c9c",
    "topicArn": "arn:aws:sns:us-east-2:664555388960:test1",
    "timestamp": "2022-09-08 15:40:57.144"
  },
  "delivery": {
    "deliveryId": "6a422437-78cc-5171-ad64-7fa3778507aa",
    "destination": "arn:aws:sqs:us-east-2:664555388960:test",
    "providerResponse": "The topic's data protection policy prohibits this message from
being delivered to <subscription arn>",
    "dwellTimeMs": 22,
    "attempts": 1,
    "statusCode": 403
  },
  "status": "FAILURE"
}

```

## Criar políticas de proteção de dados

As [políticas de proteção de dados](#) ajudam você a proteger os dados publicados em seus tópicos do Amazon SNS auditando, desidentificando (mascarando ou eliminando) e negando (bloqueando) informações confidenciais passadas entre aplicações ou Serviços da AWS. É possível usar a API da AWS, a AWS CLI, o AWS CloudFormation ou o AWS Management Console para criar políticas de proteção de dados no Amazon SNS. Somente uma política pode ser definida por tópico do Amazon SNS. Cada política de proteção de dados pode ter uma ou mais instruções de desidentificação e negação, mas somente uma instrução de auditoria.

### Tópicos

- [Criar políticas de proteção de dados para proteger os dados das mensagens \(API\)](#)
- [Criar políticas de proteção de dados para proteger os dados das mensagens \(CLI\)](#)
- [Criar políticas de proteção de dados para proteger os dados das mensagens \(CloudFormation\)](#)
- [Criar políticas de proteção de dados para proteger os dados das mensagens \(console\)](#)
- [Criar políticas de proteção de dados para proteger os dados das mensagens \(SDK\)](#)

## Criar políticas de proteção de dados para proteger os dados das mensagens (API)

O número e o tamanho dos recursos do Amazon SNS em uma conta da AWS são limitados. Para obter mais informações, consulte [Amazon Simple Notification Service endpoints and quotas](#) (Endpoints e cotas do Amazon Simple Notification Service).

### Criar políticas de proteção de dados (API da AWS)

Você pode criar uma política de proteção de dados do Amazon SNS usando a API da AWS.

Como criar uma política de proteção de dados com um tópico do Amazon SNS (API da AWS)

Use a propriedade `DataProtectionPolicy` de um tópico padrão do Amazon SNS:

- [CreateTopic](#)

Para recuperar ou criar uma política de proteção de dados para um tópico existente do Amazon SNS (API da AWS)

Chame uma das seguintes operações:

- [GetDataProtectionPolicy](#)
- [PutDataProtectionPolicy](#)

## Criar políticas de proteção de dados para proteger os dados das mensagens (CLI)

O número e o tamanho dos recursos do Amazon SNS em uma conta da AWS são limitados. Para obter mais informações, consulte [Amazon Simple Notification Service endpoints and quotas](#) (Endpoints e cotas do Amazon Simple Notification Service).

## Criar políticas de proteção de dados (AWS CLI)

Você pode criar uma política de proteção de dados do Amazon SNS usando a AWS Command Line Interface.

Como criar uma política de proteção de dados com um tópico do Amazon SNS (AWS CLI)

Use essa opção para criar uma política de proteção de dados com um tópico padrão do Amazon SNS:

- [create-topic](#)

Como recuperar ou criar uma política de proteção de dados para um tópico existente do Amazon SNS (AWS CLI)

Chame uma das seguintes operações:

- [get-data-protection-policy](#)
- [put-data-protection-policy](#)

## Criar políticas de proteção de dados para proteger os dados das mensagens (CloudFormation)

O número e o tamanho dos recursos do Amazon SNS em uma conta da AWS são limitados. Para obter mais informações, consulte [Amazon Simple Notification Service endpoints and quotas](#) (Endpoints e cotas do Amazon Simple Notification Service).

Criar políticas de proteção de dados (CloudFormation)

Você pode criar uma política de proteção de dados do Amazon SNS usando o AWS CloudFormation.

Como criar uma política de proteção de dados com um tópico do Amazon SNS (CloudFormation)

Use essa opção para criar uma política de proteção de dados com um tópico padrão do Amazon SNS:

- [AWS::SNS::Topic](#)

## Criar políticas de proteção de dados para proteger os dados das mensagens (console)

O número e o tamanho dos recursos do Amazon SNS em uma conta da AWS são limitados.

Para obter mais informações, consulte [Amazon Simple Notification Service endpoints and quotas](#) (Endpoints e cotas do Amazon Simple Notification Service).

Como criar uma política de proteção de dados com um tópico do Amazon SNS (console)

Use essa opção para criar uma política de proteção de dados com um tópico padrão do Amazon SNS.

1. Faça login no console [do Amazon SNS](#).
2. Selecione um tópico ou crie um. Para obter mais detalhes sobre como criar tópicos, consulte [Criar um tópico do Amazon SNS](#).
3. Na página Create topic (Criar tópico), na seção Details (Detalhes), selecione Standard (Padrão).
  - a. Insira um Name (Nome) para o tópico.
  - b. (Opcional) Insira um Display name (Nome para exibição) para o tópico.
4. Amplie Data protection policy (Política de proteção de dados).
5. Selecione um Configuration mode (Modo de configuração):
  - Basic (Básico): defina uma política de proteção de dados usando um menu simples.
  - Advanced (Avançado): defina uma política de proteção de dados personalizada usando JSON.
6. (Opcional) Para criar o próprio identificador de dados personalizado, expanda a seção Configuração personalizada do identificador de dados e faça o seguinte:
  - a. Insira um nome exclusivo para o identificador de dados personalizado. Os nomes de identificadores de dados personalizados aceitam caracteres alfanuméricos, sublinhado (\_) e hífen (-). São aceitos até 128 caracteres. Esse nome não pode compartilhar o mesmo nome de um [identificador de dados gerenciado](#). Para obter uma lista completa de limitações de identificadores de dados personalizados, consulte [Restrições de identificadores de dados personalizados](#).
  - b. Insira uma expressão regular (RegEx) para o identificador de dados personalizado. RegExsuporta caracteres alfanuméricos, caracteres RegEx reservados e símbolos. RegEx tem um comprimento máximo de 200 caracteres. Se RegEx for muito complicado,

o Amazon SNS falhará na chamada da API. Para obter uma lista completa das RegEx limitações, consulte [Restrições de identificadores de dados personalizados](#).

- c. (Opcional) Selecione Adicionar identificador de dados personalizado para adicionar outros identificadores de dados, conforme necessário. As políticas de proteção de dados são compatíveis com dez identificadores de dados personalizados, no máximo.
7. Selecione as declarações que você gostaria de adicionar à sua política de proteção de dados. Você pode adicionar os tipos de instrução audit (auditar), de-identify (desidentificar) (mascarar ou eliminar) e deny (negar) (bloquear) à mesma política de proteção de dados.
- a. Add audit statement (Adicionar declaração de auditoria): configure quais dados sigilosos devem ser auditados, qual porcentagem de mensagens você deseja auditar nesses dados e para onde enviar os logs de auditoria.

 Note

Somente uma declaração de auditoria é permitida por política ou tópico de proteção de dados.

- i. Selecione data identifiers (identificadores de dados) para definir os dados sigilosos que você deseja auditar.
- ii. Em Audit sample rate (Taxa de amostragem de auditoria), insira a porcentagem de mensagens a serem auditadas quanto a informações sigilosas, até no máximo 99%.
- iii. Em Audit destination (Destino de auditoria), selecione para quais Serviços da AWS enviar os resultados de constatações da auditoria e insira um nome de destino para cada AWS service (Serviço da AWS) utilizado. Você pode selecionar entre os seguintes serviços da Amazon Web Services:
  - Amazon CloudWatch — CloudWatch Logs é a solução de registro AWS padrão. Usando o CloudWatch Logs, você pode realizar análises de registros usando o Logs Insights ([veja exemplos aqui](#)) e criar métricas e alarmes. CloudWatch É nos registros que muitos serviços publicam registros, o que facilita a agregação de todos os registros usando uma única solução. Para obter informações sobre a Amazon CloudWatch, consulte o [Guia CloudWatch do usuário da Amazon](#).
  - Amazon Data Firehose — O Firehose satisfaz as demandas de streaming em tempo real para o Splunk e o OpenSearch Amazon Redshift para análises adicionais de log.

Para obter informações sobre o Amazon Data Firehose, consulte o Guia do usuário do [Amazon Data Firehose](#).

- Amazon Simple Storage Service: o Amazon S3 é um destino de log com bom custo-benefício para fins de arquivamento. Pode ser necessário manter os logs por um período de anos. Nesse caso, você pode colocar os logs no Amazon S3 para reduzir os custos. Para obter informações sobre o Amazon Simple Storage Service, consulte o [Guia do usuário do Amazon Simple Storage Service](#).

- b. Add a de-identify statement (Adicionar uma instrução de desidentificação): configure os dados confidenciais que você deseja desidentificar na mensagem, se deseja ou não mascarar ou eliminar esses dados, e as contas para interromper a entrega desses dados.
  - i. Em Data identifiers (Identificadores de dados), selecione os dados confidenciais que deseja desidentificar.
  - ii. Em Define this de-identify statement for (Definir esta instrução de desidentificação para), selecione as contas da AWS ou entidades principais do IAM às quais essa instrução de desidentificação se aplica. Você pode aplicá-la a todas as contas da AWS ou contas da AWS específicas ou a entidades do IAM (raízes, perfis ou usuários da conta) que usam IDs de conta ou ARNs de entidades do IAM. Separe vários IDs ou ARNs com uma vírgula (,).

As seguintes entidades principais do [IAM](#) são compatíveis:

- IAM account principals (Entidades principais de conta do IAM): por exemplo, `arn:aws:iam::AWS-account-ID:root`.
  - IAM role principals (Entidades principais de perfil do IAM): por exemplo, `arn:aws:iam::AWS-account-ID:role/role-name`.
  - IAM user principals (Entidades principais de usuário do IAM): por exemplo, `arn:aws:iam::AWS-account-ID:user/user-name`.
- iii. Em De-identify Option (Opção de desidentificação), selecione como deseja desidentificar os dados confidenciais. Há suporte para as seguintes opções:
    - Redact (Eliminar): remoção total dos dados. Por exemplo, e-mail: `classified@amazon.com` se torna e-mail: `.`
    - Mask (Máscara): substitui os dados por caracteres únicos. Por exemplo, e-mail: `classified@amazon.com` se torna e-mail: `*****`.
  - iv. (Opcional) Continue adicionando instruções de desidentificação conforme necessário.

- c. Add deny statement (Adicionar declaração de negação): configure quais dados sigilosos não devem percorrer o tópico e quais entidades principais não devem entregar esses dados.
  - i. Em data direction (direcionamento de dados), escolha o direcionamento das mensagens para a instrução de negação:
    - Inbound messages (Mensagens de entrada): aplique essa declaração de negação às mensagens enviadas ao tópico.
    - Outbound messages (Mensagens de saída): aplique essa declaração de negação às mensagens que o tópico entrega aos endpoints de assinatura.
  - ii. Selecione data identifiers (identificadores de dados) para definir os dados sigilosos que você deseja negar.
  - iii. Selecione as IAM principals (Entidades principais do IAM) que se aplicam a essa instrução de negação. Você pode aplicá-la a todas as contas da AWS, a Contas da AWS específicas ou a entidades do IAM (por exemplo, raízes, perfis ou usuários de conta) que usam IDs de conta ou ARNs de entidades do IAM. Separe vários IDs ou ARNs com uma vírgula (,). As seguintes entidades principais do [IAM](#) são compatíveis:
    - IAM account principals (Entidades principais de conta do IAM): por exemplo, `arn:aws:iam::AWS-account-ID:root`.
    - IAM role principals (Entidades principais de perfil do IAM): por exemplo, `arn:aws:iam::AWS-account-ID:role/role-name`.
    - IAM user principals (Entidades principais de usuário do IAM): por exemplo, `arn:aws:iam::AWS-account-ID:user/user-name`.
  - iv. (Opcional) Continue adicionando declarações de negação conforme necessário.

## Criar políticas de proteção de dados para proteger os dados das mensagens (SDK)

O número e o tamanho dos recursos do Amazon SNS em uma conta da AWS são limitados. Para obter mais informações, consulte [Amazon Simple Notification Service endpoints and quotas](#) (Endpoints e cotas do Amazon Simple Notification Service).

### Criar políticas de proteção de dados (AWS SDK)

Você pode criar uma política de proteção de dados do Amazon SNS usando o AWS SDK.

## Como criar uma política de proteção de dados com um tópico do Amazon SNS (AWS SDK)

Use as seguintes opções para criar uma política de proteção de dados com um tópico padrão do Amazon SNS:

### Java

```
/**
 * For information regarding CreateTopic see this documentation topic:
 *
 * https://docs.aws.amazon.com/code-samples/latest/catalog/javav2-sns-src-main-java-
 * com-example-sns-CreateTopic.java.html
 */

public static String createSNSTopicWithDataProtectionPolicy(SnsClient snsClient,
String topicName, String dataProtectionPolicy) {

    try {
        CreateTopicRequest request = CreateTopicRequest.builder()
            .name(topicName)
            .dataProtectionPolicy(dataProtectionPolicy)
            .build();

        CreateTopicResponse result = snsClient.createTopic(request);
        return result.topicArn();
    } catch (SnsException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
    return "";
}
```

### JavaScript

```
// Import required AWS SDK clients and commands for Node.js
import {CreateTopicCommand } from "@aws-sdk/client-sns";
import {snsClient } from "../libs/snsClient.js";

// Set the parameters
const params = { Name: "TOPIC_NAME", DataProtectionPolicy:
"DATA_PROTECTION_POLICY" };

const run = async () => {
```

```
try {
    const data = await snsClient.send(new CreateTopicCommand(params));
    console.log("Success.", data);
    return data; // For unit tests.
} catch (err) {
    console.log("Error", err.stack);
}
};
run();
```

## Como recuperar ou criar uma política de proteção de dados para um tópico existente do Amazon SNS (AWS SDK)

Use as seguintes opções para criar ou recuperar uma política de proteção de dados com um tópico padrão do Amazon SNS:

### Java

```
public static void putDataProtectionPolicy(SnsClient snsClient, String topicName,
String dataProtectionPolicy) {

    try {
        PutDataProtectionPolicyRequest request =
PutDataProtectionPolicyRequest.builder()
            .resourceArn(topicName)
            .dataProtectionPolicy(dataProtectionPolicy)
            .build();

        PutDataProtectionPolicyResponse result =
snsClient.putDataProtectionPolicy(request);
        System.out.println("\n\nStatus was " +
result.sdkHttpResponse().statusCode()
            + "\n\nTopic " + request.resourceArn()
            + " DataProtectionPolicy " + request.dataProtectionPolicy());
    } catch (SnsException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}
```

```
public static void getDataProtectionPolicy(SnsClient snsClient, String topicName) {

    try {
        GetDataProtectionPolicyRequest request =
        GetDataProtectionPolicyRequest.builder()
            .resourceArn(topicName)
            .build();

        GetDataProtectionPolicyResponse result =
        snsClient.getDataProtectionPolicy(request);

        System.out.println("\n\nStatus is " + result.sdkHttpResponse().statusCode()
            + "\n\nDataProtectionPolicy: \n\n" + result.dataProtectionPolicy());
    } catch (SnsException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}
```

## JavaScript

```
// Import required AWS SDK clients and commands for Node.js
import {PutDataProtectionPolicyCommand, GetDataProtectionPolicyCommand } from "@aws-
sdk/client-sns";
import {snsClient } from "./libs/snsClient.js";

// Set the parameters
const putParams = { ResourceArn: "TOPIC_ARN", DataProtectionPolicy:
    "DATA_PROTECTION_POLICY" };

const runPut = async () => {
    try {
        const data = await snsClient.send(new
        PutDataProtectionPolicyCommand(putParams));
        console.log("Success.", data);
        return data; // For unit tests.
    } catch (err) {
        console.log("Error", err.stack);
    }
};
runPut();

// Set the parameters
```

```
const getParams = { ResourceArn: "TOPIC_ARN" };

const runGet = async () => {
  try {
    const data = await snsClient.send(new
    GetDataProtectionPolicyCommand(getParams));
    console.log("Success.", data);
    return data; // For unit tests.
  } catch (err) {
    console.log("Error", err.stack);
  }
};

runGet();
```

## Excluir políticas de proteção de dados no Amazon SNS

É possível excluir políticas de proteção de dados do Amazon SNS usando a API da AWS, a AWS CLI, o AWS CloudFormation ou o AWS Management Console.

Para obter informações gerais sobre políticas de proteção de dados do Amazon SNS, consulte [Noções básicas sobre políticas de proteção de dados](#).

O número e o tamanho dos recursos da política de proteção de dados do Amazon SNS em uma conta da AWS são limitados. Para obter mais informações, consulte [Controle de utilização da API do Amazon SNS](#) na Referência geral da AWS.

### Tópicos

- [Excluir políticas de proteção de dados \(console\)](#)
- [Excluir uma política de proteção de dados usando uma string JSON em branco](#)
- [Excluir uma política de proteção de dados usando a AWS CLI](#)

## Excluir políticas de proteção de dados (console)

Como excluir uma política de proteção de dados gerenciados (console)

1. Faça login no [console do Amazon SNS](#).
2. Selecione o tópico que contém a política de proteção de dados que você deseja excluir.
3. Escolha Editar.

4. Expanda a seção Data protection policy (Política de proteção de dados).
5. Selecione Remove (Remover) ao lado da declaração da política de proteção de dados que você deseja remover.
6. Escolha Save changes (Salvar alterações).

## Excluir uma política de proteção de dados usando uma string JSON em branco

Você pode excluir uma política de proteção de dados atualizando-a para uma string JSON em branco.

## Excluir uma política de proteção de dados usando a AWS CLI

Você pode excluir uma política de proteção de dados usando a AWS CLI.

```
//aws sns put-data-protection-policy --resource-arn topic-arn --data-protection-policy ""
```

## Identificadores de dados

O Amazon SNS usa uma combinação de critérios e técnicas, inclusive machine learning e correspondência de padrões, para detectar dados sigilosos. Esses critérios e técnicas, coletivamente denominados identificadores de dados, podem detectar uma lista extensa e crescente de tipos de dados sigilosos para muitos países e regiões. Os identificadores de dados gerenciados do Amazon SNS oferecem tipos de dados predefinidos para proteger dados financeiros, informações pessoais de saúde (PHI) e informações de identificação pessoal (PII). Também é possível usar identificadores de dados personalizados para criar seus próprios identificadores de dados personalizados para um caso de uso específico.

### Tópicos

- [Usar identificadores de dados gerenciados no Amazon SNS](#)
- [Usar identificadores de dados personalizados no Amazon SNS](#)

## Usar identificadores de dados gerenciados no Amazon SNS

### Tópicos

- [O que são identificadores de dados gerenciados?](#)
- [Tipos de dados sigilosos: credenciais](#)

- [Tipos de dados sigilosos: dispositivos](#)
- [Tipos de dados sigilosos: financeiros](#)
- [Tipos de dados sigilosos: informações de saúde protegidas \(PHI\)](#)
- [Tipos de dados sigilosos: informações de identificação pessoal \(PII\)](#)

## O que são identificadores de dados gerenciados?

Os identificadores de dados gerenciados do Amazon SNS são projetados para detectar um tipo específico de dados sigilosos, como números de cartão de crédito, chaves de acesso secretas da AWS ou números de passaporte de determinado país ou região. Ao criar uma política de proteção de dados, você pode configurar o Amazon SNS para usar esses identificadores a fim de analisar mensagens que estão passando pelo tópico e tomar medidas quando elas forem detectadas.

O Amazon SNS pode detectar as seguintes categorias de dados sigilosos usando identificadores de dados gerenciados:

- Credenciais, como chaves privadas ou chaves de acesso secretas AWS
- Identificadores de dispositivos, como endereço IP ou MAC.
- Informações financeiras, como números de cartão de crédito
- Informações de saúde, para PHI, como seguro de saúde ou números de identificação médica.
- Informações pessoais, para PII, como carteiras de habilitação ou números de previdência social.

Em cada categoria, o Amazon SNS pode detectar vários tipos de dados sigilosos. Os tópicos dessa seção listam e descrevem cada tipo e todos os requisitos relevantes para detectá-lo. Para cada tipo, eles também indicam o identificador exclusivo (ID) do identificador de dados gerenciados projetado para detectar os dados. Ao criar uma política de proteção de dados, você pode usar esse ID para incluir o identificador de dados gerenciados a ser detectado pela proteção de dados de mensagens.

### Requisitos de palavras-chave

Para detectar determinados tipos de dados sigilosos, o Amazon SNS verifica as palavras-chave próximas aos dados. Se esse for o caso de determinado tipo de dado, um tópico subsequente nessa seção indicará requisitos específicos de palavras-chave para esses dados.

As palavras-chave não diferenciam maiúsculas de minúsculas. Além disso, se uma palavra-chave contiver um espaço, o Amazon SNS faz automaticamente a correspondência com as variações de palavras-chave que não contêm o espaço ou contêm um sublinhado ( \_ ) ou um hífen ( - ) em vez do

espaço. Em determinados casos, o Amazon SNS também estende ou abrevia uma palavra-chave para abordar variações comuns da palavra-chave.

### Identificadores de dados gerenciados pelo Amazon SNS para tipos de dados sigilosos

A tabela a seguir lista e descreve os tipos de credencial, dispositivo, informações financeiras, médicas e pessoais de saúde (PHI) que o Amazon SNS pode detectar usando identificadores de dados gerenciados. Eles são uma adição a determinados tipos de dados que também podem ser qualificados como informações de identificação pessoal (PII).

Os identificadores de dados dependentes da região exigem o nome do identificador com um hífen e os códigos de duas letras (ISO 3166-1 alfa-2). Por exemplo, DriversLicense -US.

Identificador	Categoria	Países/idiomas
BankAccountNumber	Financeiro	DE, ES, FR, GB, IT
CepCode	Pessoal	BR
Cnpj	Pessoal	BR
CpfCode	Pessoal	BR
DriversLicense	Pessoal	AT, AU, BE, BG, CA, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IT, LT, LU, LV, MT, NL, PL, PT, RO, SE, SI, SK, US
DrugEnforcementAgencyNumber	Integridade	EUA
ElectoralRollNumber	Pessoal	GB
HealthInsuranceCardNumber	Integridade	UE
HealthInsuranceClaimNumber	Integridade	EUA
HealthInsuranceNumber	Integridade	FR
HealthcareProcedureCode	Integridade	EUA

Identificador	Categoria	Países/idiomas
IndividualTaxIdentificationNumber	Pessoal	EUA
InseeCode	Pessoal	FR
MedicareBeneficiaryNumber	Integridade	EUA
NationalDrugCode	Integridade	EUA
NationalIdentificationNumber	Pessoal	DE, ES, IT
NationalInsuranceNumber	Pessoal	GB
NationalProviderId	Integridade	EUA
NhsNumber	Integridade	GB
NieNumber	Pessoal	ES
NifNumber	Pessoal	ES
PassportNumber	Pessoal	CA, DE, ES, FR, GB, IT, US
PermanentResidenceNumber	Pessoal	CA
PersonalHealthNumber	Integridade	CA
PhoneNumber	Pessoal	BR, DE, ES, FR, GB, IT, US
PostalCode	Pessoal	CA
RgNumber	Pessoal	BR
SocialInsuranceNumber	Pessoal	CA
Ssn	Pessoal	ES, US
TaxId	Pessoal	DE, ES, FR, GB
ZipCode	Pessoal	EUA

## Identificadores compatíveis que são independentes do idioma/região

Identificador	Categoria
Endereço	Pessoal
AwsSecretKey	Credenciais
CreditCardExpiration	Financeiro
CreditCardNumber	Financeiro
CreditCardSecurityCode	Financeiro
EmailAddress	Pessoal
IpAddress	Pessoal
LatLong	Pessoal
Nome	Pessoal
OpenSshPrivateKey	Credenciais
PgpPrivateKey	Credenciais
PkcsPrivateKey	Credenciais
PuttyPrivateKey	Credenciais
VehicleIdentificationNumber	Pessoal

### Tipos de dados sigilosos: credenciais

A tabela a seguir lista e descreve os tipos de credencial que o Amazon SNS pode detectar usando identificadores de dados gerenciados.

Tipo de detecção	ID do identificador de dados gerenciados	Palavra-chave obrigatória	Países e regiões
Chave de acesso secreta da AWS	AwsSecretKey	aws_secret_access_key, credentials, secret access key, secret key, set-awscredential	Any
Chave privada OpenSSH	OpenSshPrivateKey	Não	Any
Chave privada PGP	PgpPrivateKey	Não	Any
Chave privada do padrão de criptografia de chave pública (PKCS)	PkcsPrivateKey	Não	Any
Chave privada PuTTY	PuttyPrivateKey	Não	Any

## ARNs de identificadores de dados para tipos de dados de credencial

Veja a seguir os nomes de recurso da Amazon (ARNs) para os identificadores de dados que você pode adicionar às suas políticas de proteção de dados.

### ARNs de identificadores de dados de credenciais

```
arn:aws:dataprotection: :aws:data-identifier/ AwsSecretKey
```

```
arn:aws:dataprotection: :aws:data-identifier/ OpenSshPrivateKey
```

```
arn:aws:dataprotection: :aws:data-identifier/ PgpPrivateKey
```

```
arn:aws:dataprotection: :aws:data-identifier/ PkcsPrivateKey
```

```
arn:aws:dataprotection: :aws:data-identifier/ PuttyPrivateKey
```

## Tipos de dados sigilosos: dispositivos

A tabela a seguir lista e descreve os tipos de identificador de dispositivos que o Amazon SNS pode detectar usando identificadores de dados gerenciados.

Tipo de detecção	ID do identificador de dados gerenciados	Palavra-chave obrigatória	Países e regiões
Endereço IP	IpAddress	Não	Any

### ARNs de identificadores de dados para tipos de dados de dispositivos

Veja a seguir os nomes de recurso da Amazon (ARNs) para os identificadores de dados que você pode adicionar às suas políticas de proteção de dados.

#### ARN de identificador de dados de dispositivos

```
arn:aws:dataprotection: :aws:data-identifier/ IpAddress
```

## Tipos de dados sigilosos: financeiros

A tabela a seguir lista e descreve os tipos de informações financeiras que o Amazon SNS pode detectar usando identificadores de dados gerenciados.

Tipo de detecção	ID do identificador de dados gerenciados	Palavra-chave obrigatória	Mais informações	Países e regiões
Número de conta bancária	BankAccountNumber  BankAccountNumber-US	Sim, consulte <a href="#">Palavras-chave para números de conta bancária.</a>	Isso inclui: números de contas bancárias internacionais (IBANs) compostos de até 34 caracteres alfanuméricos	França, Alemanha, Itália, Espanha, Reino Unido

Tipo de detecção	ID do identificador de dados gerenciados	Palavra-chave obrigatória	Mais informações	Países e regiões
			icos, incluindo elementos como o código do país.	
Data de validade do cartão de crédito	CreditCardExpiration	exp d, exp m, exp y, expiração, expirar	–	Any
Dados da faixa magnética do cartão de crédito	CreditCardMagneticStripe	Sim, incluindo : card data, iso7813, mag, magstripe, stripe, swipe.	Isso inclui as faixas 1 e 2.	Any

Tipo de detecção	ID do identificador de dados gerenciados	Palavra-chave obrigatória	Mais informações	Países e regiões
Números de cartão de crédito	CreditCardNumber	account number, american express, amex, bank card, card, card num, card number, cc #, ccn, check card, credit, credit card#, dankort, debit, debit card, diners club, discover, electron, elo verification code, japanese card bureau, jcb, mastercard, mc, pan, payment account number, payment card number, pcn, union pay, visa	A detecção exige que os dados sejam uma sequência de 13 a 19 dígitos que siga a fórmula de cheque de Luhn e use um prefixo de número de cartão padrão para qualquer um dos seguintes tipos de cartão de crédito: American Express, Dankort, Diner's Club, Discover, Electron, Japanese Card Bureau (JCB), Mastercard e Visa (link sobrescrito abaixo de 1). UnionPay	Any

Tipo de detecção	ID do identificador de dados gerenciados	Palavra-chave obrigatória	Mais informações	Países e regiões
Código de verificação do cartão de crédito	CreditCardSecurityCode	id do cartão, código de identificação do cartão, número de identificação do cartão, código de segurança do cartão, código de validação do cartão, número de validação do cartão, dados de verificação do cartão, valor de verificação do cartão, cvc, cvc2, cvv, cvv2, código de verificação elo	–	Any

1.

O Amazon SNS não relata ocorrências das seguintes sequências, que os emissores de cartão de crédito reservaram para testes públicos:

122000000000003, 2222405343248877, 2222990905257051, 2223007648726984, 2223577120017656, 30569309025904, 34343434343434, 3528000700000000, 3530111333300000, 3566002020360505, 36148900647913, 36700102000000, 371449635398431, 378282246310005, 378734493671000, 38520000023237, 401288888881881, 4111111111111111, 42222222222222, 4444333322221111, 4462030000000000, 4484070000000000, 49118300000000, 4917300800000000, 4917610000000000, 4917610000000000003, 5019717010103742, 5105105105105100, 5111010030175156, 5185540810000019, 5200828282828210, 5204230080000017,

5204740009900014, 5420923878724339, 5454545454545454, 5455330760000018, 5506900490000436, 5506900490000444, 5506900510000234, 5506920809243667, 5506922400634930, 5506927427317625, 5553042241984105, 5555553753048194, 555555555554444, 5610591081018250, 6011000990139424, 6011000400000000, 6011111111111117, 630490017740292441, 630495060000000000, 6331101999990016, 6759649826438453, 6799990100000000019 e 76009244561.

## Palavras-chave para números de conta bancária

Use as palavras-chave a seguir para detectar números de conta bancária internacional (IBANs) compostos de até 34 caracteres alfanuméricos, inclusive elementos como o código do país.

País ou região	Palavras-chave			
França	account code, account number, accountno #, accountnu mber#, bban, code bancaire, compte bancaire, customer account id, customer account number, customer bank account id, iban, numéro de compte			
Alemanha	account code, account number, accountno #, accountnu mber#, bankleitz ahl, bban, customer			

País ou região	Palavras-chave			
	account id, customer account number, customer bank account id, geheimzahl, iban, kartenum mer, kontonumm er, kreditkar tennummer, sepa			
Itália	account code, account number, accountno #, accountnu mber#, bban, codice bancario, conto bancario, customer account id, customer account number, customer bank account id, iban, numero di conto			

País ou região	Palavras-chave			
Espanha	account code, account number, accountno #, accountnu mber#, bban, código cuenta, código cuenta bancaria, cuenta cliente id, customer account ID, customer account number, customer bank account id, iban, número cuenta bancaria cliente, número cuenta cliente			
Reino Unido	account code, account number, accountno #, accountnu mber#, bban, customer account id, customer account number, customer bank account id, iban, sepa			

País ou região	Palavras-chave			
EUA	bank account, bank acct, checking account, checking acct, deposit account, deposit acct, savings account, savings acct, chequing account, chequing acct			

## ARNs de identificadores de dados para tipos de dados financeiros

Veja a seguir os nomes de recurso da Amazon (ARNs) para os identificadores de dados que você pode adicionar às suas políticas de proteção de dados.

### ARNs de identificadores de dados financeiros

```
arn:aws:dataprotection: :aws:data-identifier/ -DE BankAccountNumber
```

```
arn:aws:dataprotection: :aws:data-identifier/ -ES BankAccountNumber
```

```
arn:aws:dataprotection: :aws:data-identifier/ -FR BankAccountNumber
```

```
arn:aws:proteção de dados: :aws:data-identifier/ -GB BankAccountNumber
```

```
arn:aws:dataprotection: :aws:data-identifier/ -IT BankAccountNumber
```

```
arn:aws:dataprotection: :aws:data-identifier/ -US BankAccountNumber
```

```
arn:aws:dataprotection: :aws:data-identifier/ CreditCardExpiration
```

```
arn:aws:dataprotection: :aws:data-identifier/ CreditCardNumber
```



Tipo de detecção	ID do identificador de dados gerenciados	Palavra-chave obrigatória	Países e regiões
		da conta médica, numero conto medico, numéro d'assurance maladie, numéro de carte d'assurance, numéro de compte medical, número de cuenta médica, número de seguro de salud, número de tarjeta de seguro, sairaanhoitokortin, sairausvakuutuskortti, sairausvakuutusnumero, sjukförsäkring nummer, sjukförsäkringskort, suomi ehic-numero, tarjeta de salud, terveyskortti, tessera sanitaria assicurazione numero, versicherungsnummer	
Health Insurance Claim Number (HICN)	HealthInsuranceClaimNumber	health insurance claim number, hic no, hic no., hic number, hic#, hicn, hicn#., hicno#	EUA
Número de identificação médica ou do seguro de saúde	HealthInsuranceNumber	carte d'assuré social, carte vitale, insurance card	FR

Tipo de detecção	ID do identificador de dados gerenciados	Palavra-chave obrigatória	Países e regiões
Código do Healthcare and Common Procedure Coding System (HCPCS)	HealthcareProcedureCode	current procedural terminology, hcpcs, healthcare common procedure coding system	EUA
Número do beneficiário do Medicare (MBN)	MedicareBeneficiaryNumber	mbi, medicare beneficiary	EUA
National Drug Code (NDC)	NationalDrugCode	national drug code, ndc	EUA
National Provider Identifier (NPI)	NationalProviderId	hipaa, n.p.i, national provider, npi	EUA
Número do Serviço Nacional de Saúde (NHS)	NhsNumber	national health service, NHS	GB
Personal Health Number (PHN)	PersonalHealthNumber	canada healthcare number, msp number, personal healthcare number, phn, soins de santé	CA

### Palavras-chave para números de identificação médica e do seguro de saúde

Para detectar vários tipos de número de identificação médica e de seguro de saúde, o Amazon SNS exige que uma palavra-chave esteja próxima aos números. Isso inclui números do Cartão Europeu de Seguro de Doença (UE, Finlândia), números de seguro de saúde (França), identificadores de beneficiários do Medicare (EUA), números de seguro nacional (Reino Unido), números do NHS (Reino Unido) e números pessoais de saúde (Canadá).

A tabela a seguir lista as palavras-chave que o Amazon SNS reconhece para países e regiões específicos.

País ou região	Palavras-chave
Canadá	Canada healthcare number, msp number, personal healthcare number, phn, soins de santé
UE	assicurazione sanitaria numero, carta assicurazione numero, carte d'assurance maladie, carte européenne d'assurance maladie, ceam, ehic, ehic#, finlandehicnumber#, gesundheitskarte, hälsokort, health card, health card number, health insurance card, health insurance number, insurance card number, krankenversicherungskarte, krankenversicherungnummer, medical account number, numero conto medico, numéro d'assurance maladie, numéro de carte d'assurance, numéro de compte medical, número de cuenta médica, número de seguro de salud, número de tarjeta de seguro, sairaanhoitokortin, sairausvaakuuskortti, sairausvakuutusnumero, sjukförsäkring nummer, sjukförsäkringskort, suomi ehic-numero, tarjeta de salud, terveyskortti, tessera sanitaria assicurazione numero, versicherungsnummer
Finlândia	ehic, ehic#, finland health insurance card, finlandehicnumber#, finska sjukförsäkringskort, hälsokort, health card, health card number, health insurance card, health insurance number, sairaanhoitokortin, sairaanhoitokortin, sairausvaakuuskortti, sairausvakuutusnumero, sjukförsäkring nummer, sjukförsäkringskort

País ou região	Palavras-chave
	t, suomen sairausvakuutuskortti, suomi ehic-numero, terveyskortti
França	carte d'assuré social, carte vitale, insurance card
Reino Unido	serviço nacional de saúde, NHS
EUA	mbi, medicare beneficiary

ARNs identificadores de dados para tipos de dados de informações de saúde protegidas (PHI)

Veja a seguir os nomes de recurso da Amazon (ARNs) que podem ser usados em políticas de proteção de dados de PHI.

#### ARNs de identificadores de dados de PHI

arn:aws:dataprotection: :aws:data-identifier/ -US DrugEnforcementAgencyNumber

arn:aws:dataprotection: :aws:data-identifier/ -US HealthcareProcedureCode

arn:aws:dataprotection: :aws:data-identifier/ -EU HealthInsuranceCardNumber

arn:aws:dataprotection: :aws:data-identifier/ -US HealthInsuranceClaimNumber

arn:aws:dataprotection: :aws:data-identifier/ -FR HealthInsuranceNumber

arn:aws:dataprotection: :aws:data-identifier/ -US MedicareBeneficiaryNumber

arn:aws:dataprotection: :aws:data-identifier/ -US NationalDrugCode

arn:aws:proteção de dados: :aws:data-identifier/ -GB NationalInsuranceNumber

arn:aws:dataprotection: :aws:data-identifier/ -US NationalProviderId

arn:aws:proteção de dados: :aws:data-identifier/ -GB NhsNumber

arn:aws:dataprotection: :aws:data-identifier/ -CA PersonalHealthNumber

## Tipos de dados sigilosos: informações de identificação pessoal (PII)

A tabela a seguir lista e descreve os tipos de informações de identificação pessoal (PII) que o Amazon SNS pode detectar usando identificadores de dados gerenciados.

Tipo de detecção	ID do identificador de dados gerenciados	Palavra-chave obrigatória	Mais informações	Países e regiões
Datas de nascimento	DateOfBirth	dob, date of birth, birthdate, birth date, birthday, b-day, bday	O suporte inclui a maioria dos formatos de data, como todos os dígitos e combinações de dígitos e nomes de meses. Os componentes de data podem ser separados por espaços, barras (/) ou hífen (-).	Any
Código de endereçamento postal (CEP)	CepCode	cep, código de endereçamento postal, código de endereçamento postal	–	Brasil
Cadastro Nacional da Pessoa Jurídica (CNPJ)	Cnpj	cadastro nacional da pessoa jurídica, cadastro nacional da pessoa jurídica, cnpj	–	Brasil

Tipo de detecção	ID do identificador de dados gerenciados	Palavra-chave obrigatória	Mais informações	Países e regiões
Cadastro de Pessoa Física (CPF)	CpfCode	Cadastro de pessoas físicas, cadastro de pessoas físicas, cadastro de pessoa física, cadastro de pessoa física, cpf	–	Brasil

Tipo de detecção	ID do identificador de dados gerenciados	Palavra-chave obrigatória	Mais informações	Países e regiões
Número de identificação da carteira de habilitação	DriversLicense	Sim, consulte <a href="#">Palavras-chave para números de identificação da carteira de habilitação</a> .	–	Austrália, Áustria, Bélgica, Bulgária, Canadá, Croácia, Chipre, República Tcheca, Dinamarca, Estônia, Finlândia, França, Alemanha, Grécia, Hungria, Irlanda, Itália, Letônia, Lituânia, Luxemburgo, Malta, Holanda, Polônia, Portugal, Romênia, Eslováquia, Eslovênia, Espanha, Suécia, Reino Unido, EUA

Tipo de detecção	ID do identificador de dados gerenciados	Palavra-chave obrigatória	Mais informações	Países e regiões
Número de registro eleitoral	Electoral RollNumber	electoral#, electoral #, electoralnumber, electoral number, electoralroll#, electoral roll#, electoral roll #, electoral roll no., electoral roll number, electoralrollno	–	Reino Unido
Identificação do contribuinte individual	Individual TaxIdentification Number	Sim, consulte <a href="#">Palavras-chave para identificação do contribuinte e números de referência.</a>	–	EUA
Instituto Nacional de Estatística e Estudos Econômicos (INSEE)	InseeCode	Sim, consulte <a href="#">Palavras-chave para números de identificação nacional.</a>	–	França

Tipo de detecção	ID do identificador de dados gerenciados	Palavra-chave obrigatória	Mais informações	Países e regiões
Número de identificação nacional	NationalIdentificationNumber	Sim, consulte <a href="#">Palavras-chave para números de identificação nacional.</a>	Isso inclui identificadores do Documento Nacional de Identidade (DNI) (Espanha), códigos fiscais (Itália) e números de carteira de identidad e nacional (Alemanha).	Alemanha, Itália, Espanha
Número do Seguro Nacional (NINO)	NationalInsuranceNumber	insurance no., insurance number, insurance #, national insurance number, nationalinsurance#, nationalinsurancenum, nin, nino	–	Reino Unido
Número de identidad de extranjero (NIE)	NieNumber	Sim, consulte <a href="#">Palavras-chave para identificação do contribuinte e números de referência.</a>	–	Espanha

Tipo de detecção	ID do identificador de dados gerenciados	Palavra-chave obrigatória	Mais informações	Países e regiões
Número de Identificación Fiscal (NIF)	NifNumber	Sim, consulte <a href="#">Palavras-chave para identificação do contribuinte e números de referência.</a>	–	Espanha
Número de passaporte	PassportNumber	Sim, consulte <a href="#">Palavras-chave para números de passaporte.</a>	–	Canadá, França, Alemanha, Itália, Espanha, Reino Unido, EUA

Tipo de detecção	ID do identificador de dados gerenciados	Palavra-chave obrigatória	Mais informações	Países e regiões
Número de residência permanente	Permanent Residence Number	carte résident permanent , numéro carte résident permanent, numéro résident permanent , permanent resident card, permanent resident card number, permanent resident no, permanent resident no., permanent resident number, pr no, pr no., pr non, pr number, résident permanent no., résident permanent non	–	Canadá

Tipo de detecção	ID do identificador de dados gerenciados	Palavra-chave obrigatória	Mais informações	Países e regiões
Número de telefone	PhoneNumber	<p>Brasil: as palavras-chave também incluem: cel, celular, fone, móvel, número residencial, numero residencial, telefone</p> <p>Outros: celular, contato, fax, número de fax, celular, fone, número de fone, telefone, número de telefone</p>	Isso inclui números de chamada gratuita nos EUA e números de fax. Se uma palavra-chave estiver próxima dos dados, o número não precisará incluir o código do país. Se uma palavra-chave estiver próxima dos dados, o número não precisará incluir o código do país.	Brasil, Canadá, França, Alemanha, Itália, Espanha, Reino Unido, EUA
CEP	PostalCode	No	–	Canadá
Registro Geral (RG)	RgNumber	Sim, consulte <a href="#">Palavras-chave para números de identificação nacional</a> .	–	Brasil

Tipo de detecção	ID do identificador de dados gerenciados	Palavra-chave obrigatória	Mais informações	Países e regiões
Número do Seguro Social (SIN)	SocialInsuranceNumber	canadian id, número d'assurance sociale, social insurance number, sin	–	Canadá
Número da Previdência Social (SSN)	Ssn	Espanha: número de la seguridad social, social security no., social security no. número de la seguridad social, social security number, socialsecurityno#, ssn, ssn#  EUA: previdência social, ss#, ssn	–	Espanha, EUA
Identificação do contribuinte ou número de referência	TaxId	Sim, consulte <a href="#">Palavras-chave para identificação do contribuinte e números de referência.</a>	Isso inclui TIN (França); Steueridentifikationsnummer (Alemanha); CIF (Espanha) e TRN, UTR (Reino Unido).	França, Alemanha, Espanha, Reino Unido

Tipo de detecção	ID do identificador de dados gerenciados	Palavra-chave obrigatória	Mais informações	Países e regiões
Código postal dos EUA	ZipCode	zip code, zip+4	–	EUA
Endereço postal	Address	No	Embora uma palavra-chave não seja necessária, a detecção exige que o endereço inclua o nome de uma cidade ou local e um CEP.	Austrália, Canadá, França, Alemanha, Itália, Espanha, Reino Unido, EUA
Endereço de correio eletrônico	EmailAddress	email, endereço de email, e-mail, endereço de e-mail	–	Any

Tipo de detecção	ID do identificador de dados gerenciados	Palavra-chave obrigatória	Mais informações	Países e regiões
Coordenadas do sistema de posicionamento global (GPS)	LatLong	coordinate, coordinates, lat long, latitude longitude, location, position	<p>O Amazon SNS poderá detectar coordenadas de GPS se as coordenadas de latitude e longitude estiverem armazenadas como um par e estiverem no formato de graus decimais (DD), por exemplo, 41.948614,-87.655311.</p> <p>O suporte não inclui coordenadas no formato graus, decimais, minutos (DDM), por exemplo, 41°56.9168'N 87°39.3187'W, ou no formato graus, minutos, segundos (DMS), por exemplo, 41°56'55.0104"N 87°39'19.1196"W.</p>	Any

Tipo de detecção	ID do identificador de dados gerenciados	Palavra-chave obrigatória	Mais informações	Países e regiões
Nome completo	Name	No	O Amazon SNS consegue detectar somente nomes completos. O suporte é limitado aos conjuntos de caracteres latinos.	Any
Número de identificação de veículo (VIN)	VehicleIdentificationNumber	Fahrgeste llnummer, niv, numarul de identificare, numarul seriei de sasiu, serie sasiu, numer VIN, Número de Identificação do Veículo, Número de Identificación de Automóvil es, numéro d'identification du véhicule, vehicle identification number, vin, VIN numerus	O Amazon SNS pode detectar VINs que consistem em uma sequência de 17 caracteres e aderem aos padrões ISO 3779 e 3780. Esses padrões foram projetados para uso em todo o mundo.	Any

## Palavras-chave para números de identificação da carteira de habilitação

Para detectar vários tipos de número de identificação da carteira de habilitação, o Amazon SNS exige que uma palavra-chave esteja próxima aos números. A tabela a seguir lista as palavras-chave que o Amazon SNS reconhece para países e regiões específicos.

País ou região	Palavras-chave
Austrália	dl# dl:, dl :, dlno# driver licence, driver license, driver permit, drivers lic., drivers licence, driver's licence, drivers license, driver's license, drivers permit, driver's permit, drivers permit number, driving licence, driving license, driving permit
Áustria	führerschein, fuhrerschein, führerschein republik österreich, fuhrerschein republik osterreich
Bélgica	fuehrerschein, fuehrerschein- nr, fuehrerscheinnummer, fuhrerschein, führerschein, fuhrerschein- nr, führerschein- nr, fuhrerscheinnummer, führerscheinnummer, numéro permis conduire, permis de conduire, rijbewijs, rijbewijsnummer
Bulgária	превозно средство, свидетелство за управление на моторно, свидетелство за управление на мпс, сумпс, шофьорска книжка
Canadá	dl#, dl:, dlno#, driver licence, driver licences, driver license, driver licenses, driver permit, drivers lic., drivers licence, driver's licence, drivers licences, driver's licences, drivers license, driver's license, drivers licenses, driver's licenses, drivers permit, driver's permit,

País ou região	Palavras-chave
	drivers permit number, driving licence, driving license, driving permit, permis de conduire
Croácia	vozačka dozvola
Chipre	άδεια οδήγησης
República Tcheca	číslo licence, číslo licence řidiče, číslo řidičského o průkazu, ovladače lic., povolení k jízdě, povolení řidiče, řidiči povolení, řidičský průkaz, řidičský průkaz
Dinamarca	kørekort, kørekortnummer
Estônia	juhi litsentsi number, juhiloa number, juhiluba, juhiluba number
Finlândia	ajokortin numero, ajokortti, förare lic., körkort, körkort nummer, kuljettaja lic., permis de conduire
França	permis de conduire
Alemanha	fuehrerschein, fuehrerschein- nr, fuehrerscheinnummer, fuhrerschein, fuhrerschein, fuhrerschein- nr, fuhrerschein- nr, fuhrerscheinnummer, fuhrerscheinnummer
Grécia	δεια οδήγησης, adeia odigisis
Hungria	illesztőprogramok lic, jogosítvány, jogsí, licencszám, vezető engedély, vezetői engedély
Irlanda	ceadúnas tiomána
Itália	patente di guida, patente di guida numero, patente guida, patente guida numero

País ou região	Palavras-chave
Letônia	autovadītāja apliecība, licences numurs, vadītāja apliecība, vadītāja apliecības numurs, vadītāja atļauja, vadītāja licences numurs, vadītāji lic.
Lituânia	vairuotojo pažymėjimas
Luxemburgo	fahrerlaubnis, führungsschein
Malta	licenzja tas-sewqan
Holanda	permis de conduire, rijbewijs, rijbewijsnummer
Polônia	numer licencyjny, prawo jazdy, zezwolenie na prowadzenie
Portugal	carta de condução, carteira de habilitação, carteira de motorist, carteira habilitação, carteira motorist, licença condução, licença de condução, número de licença, número licença, permissão condução, permissão de condução
Romênia	numărul permisului de conducere, permis de conducere
Eslováquia	číslo licencie, číslo vodičského preukazu, ovládače lic., povolenia vodičov, povolenie jazdu, povolenie na jazdu, povolenie vodiča, vodičský preukaz
Eslovênia	vozniško dovoljenje

País ou região	Palavras-chave
Espanha	carnet conductor, el carnet de conductor, licencia conductor, licencia de manejo, número carnet conductor, número de carnet de conductor, número de permiso conductor, número de permiso de conductor, número licencia conductor, número permiso conductor, permiso conducción, permiso conductor, permiso de conducción
Suécia	ajokortin numero, dlno# ajokortti, drivere lic., förare lic., körkort, körkort nummer, körkortsn ummer, kuljettajat lic.
Reino Unido	dl#, dl:, dlno#, driver licence, driver licences, driver license, driver licenses, driver permit, drivers lic., drivers licence, driver's licence, drivers licences, driver's licences, drivers license, driver's license, drivers licenses, driver's licenses, drivers permit, driver's permit, drivers permit number, driving licence, driving license, driving permit
EUA	dl#, dl:, dlno#, driver licence, driver licences, driver license, driver licenses, driver permit, drivers lic., drivers licence, driver's licence, drivers licences, driver's licences, drivers license, driver's license, drivers licenses, driver's licenses, drivers permit, driver's permit, drivers permit number, driving licence, driving license, driving permit

## Palavras-chave para números de identificação nacional

Para detectar vários tipos de número de identificação nacional, o Amazon SNS exige que uma palavra-chave esteja próxima aos números. Isso inclui identificadores do Documento Nacional

de Identidad (DNI) (Espanha), códigos do Instituto Nacional de Estatística e Estudos Econômicos (INSEE) da França, números da carteira de identidade nacional alemã e números do Registro Geral (RG) (Brasil).

A tabela a seguir lista as palavras-chave que o Amazon SNS reconhece para países e regiões específicos.

País ou região	Palavras-chave
Brasil	registro geral, rg
França	assurance sociale, carte nationale d'identit é, cni, code sécurité sociale, French social security number, fssn#, insee, insurance number, national id number, nationalid#, numéro d'assurance, sécurité sociale, sécurité sociale non., sécurité sociale numéro, social, social security, social security number, socialsecuritynumber, ss#, ssn, ssn#
Alemanha	ausweisnummer, id number, identification number, identity number, insurance number, personal id, personalausweis
Itália	codice fiscal, dati anagrafici, ehic, health card, health insurance card, p. iva, partita i.v.a., personal data, tax code, tessera sanitaria
Espanha	dni, dni#, dninúmero#, documento nacional de identidad, identidad único, identidadúnico#, insurance number, national identification number, national identity, nationalid#, nationali dno#, número nacional identidad, personal identification number, personal identity no, unique identity number, uniqueid#

## Palavras-chave para números de passaporte

Para detectar vários tipos de número de passaporte, o Amazon SNS exige que uma palavra-chave esteja próxima aos números. A tabela a seguir lista as palavras-chave que o Amazon SNS reconhece para países e regiões específicos.

País ou região	Palavras-chave
Canadá	passport, passport#, passport, passport#, passportno, passportno#
França	numéro de passeport, passeport, passeport #, passeport #, passeportn °, passeport n °, passeportNon, passeport non
Alemanha	ausstellungsdatum, ausstellungsort, geburtsdatum, passport, passports, reisepass, reisepassnr, reisepassnummer
Itália	italian passport number, numéro passeport , numéro passeport italien, passaporto, passaporto italiana, passaporto numero, passport number, repubblica italiana passaporto
Espanha	españa pasaporte, libreta pasaporte, número pasaporte, pasaporte, passport, passport book, passport no, passport number, spain passport
Reino Unido	passport #, passeport n °, passeportNon, passeport non, passeportn °, passport #, passport no, passport number, passport#, passportid
EUA	passport, travel document

## Palavras-chave para identificação do contribuinte e números de referência

Para detectar vários tipos de número de referência e identificação de contribuinte, o Amazon SNS exige que uma palavra-chave esteja próxima aos números. A tabela a seguir lista as palavras-chave que o Amazon SNS reconhece para países e regiões específicos.

País ou região	Palavras-chave
Brasil	cadastro de pessoa física, cadastro de pessoa física, cadastro de pessoas físicas, cadastro de pessoas físicas, cadastro nacional da pessoa jurídica, cadastro nacional da pessoa jurídica, cnpj, cpf
França	numéro d'identification fiscale, tax id, tax identification number, tax number, tin, tin#
Alemanha	identifikationsnummer, steuer id, steueridentifikationsnummer, steuernummer, tax id, tax identification number, tax number
Espanha	cif, cif número, cifnúmero#, nie, nif, número de contribuyente, número de identidad de extranjero, número de identificación fiscal, número de impuesto corporativo, personal tax number, tax id, tax identification number, tax number, tin, tin#
Reino Unido	paye, tax id, tax id no., tax id number, tax identification, tax identification#, tax no., tax number, tax reference, tax#, taxid#, temporary reference number, tin, trn, unique tax reference, unique taxpayer reference, utr
EUA	número de identificação de contribuinte individual, itin, i.t.i.n.

## ARNs do identificador de dados de informações de identificação pessoal (PII)

A tabela a seguir lista os nomes de recurso da Amazon (ARNs) para os identificadores de dados que você pode adicionar às suas políticas de proteção de dados.

### ARNs de identificadores de dados de PII

arn:aws:dataprotection::aws:data-identifier/Address

arn:aws:proteção de dados: :aws:data-identifier/ -BR CepCode

arn:aws:dataprotection::aws:data-identifier/Cnpj-BR

arn:aws:proteção de dados: :aws:data-identifier/ -BR CpfCode

arn:aws:dataprotection: :aws:data-identifier/ DateOfBirth

arn:aws:dataprotection: :aws:data-identifier/ -AT DriversLicense

arn:aws:dataprotection: :aws:data-identifier/ -AU DriversLicense

arn:aws:dataprotection: :aws:data-identifier/ -BE DriversLicense

arn:aws:dataprotection: :aws:data-identifier/ -BG DriversLicense

arn:aws:dataprotection: :aws:data-identifier/ -CA DriversLicense

arn:aws:dataprotection: :aws:data-identifier/ -CY DriversLicense

arn:aws:dataprotection: :aws:data-identifier/ -CZ DriversLicense

arn:aws:dataprotection: :aws:data-identifier/ -DE DriversLicense

arn:aws:dataprotection: :aws:data-identifier/ -DK DriversLicense

arn:aws:dataprotection: :aws:data-identifier/ -EE DriversLicense

arn:aws:dataprotection: :aws:data-identifier/ -ES DriversLicense

arn:aws:dataprotection: :aws:data-identifier/ FI DriversLicense

arn:aws:dataprotection: :aws:data-identifier/ -FR DriversLicense

## ARNs de identificadores de dados de PII

arn:aws:proteção de dados: :aws:data-identifier/ -GB DriversLicense

arn:aws:dataprotection: :aws:data-identifier/ -GR DriversLicense

arn:aws:dataprotection: :aws:data-identifier/ -HR DriversLicense

arn:aws:dataprotection: :aws:data-identifier/ -HU DriversLicense

arn:aws:dataprotection: :aws:data-identifier/ -IE DriversLicense

arn:aws:dataprotection: :aws:data-identifier/ -IT DriversLicense

arn:aws:dataprotection: :aws:data-identifier/ -LT DriversLicense

arn:aws:dataprotection: :aws:data-identifier/ -LU DriversLicense

arn:aws:dataprotection: :aws:data-identifier/ -LV DriversLicense

arn:aws:dataprotection: :aws:data-identifier/ -MT DriversLicense

arn:aws:dataprotection: :aws:data-identifier/ -NL DriversLicense

arn:aws:proteção de dados: :aws:data-identifier/ -PL DriversLicense

arn:aws:dataprotection: :aws:data-identifier/ -PT DriversLicense

arn:aws:dataprotection: :aws:data-identifier/ -RO DriversLicense

arn:aws:proteção de dados: :aws:data-identifier/ -SE DriversLicense

arn:aws:dataprotection: :aws:data-identifier/ -SI DriversLicense

arn:aws:dataprotection: :aws:data-identifier/ -SK DriversLicense

arn:aws:dataprotection: :aws:data-identifier/ -US DriversLicense

arn:aws:proteção de dados: :aws:data-identifier/ -GB ElectoralRollNumber

arn:aws:dataprotection: :aws:data-identifier/ EmailAddress

## ARNs de identificadores de dados de PII

arn:aws:dataprotection: :aws:data-identifier/ -US IndividualTaxIdentificationNumber

arn:aws:dataprotection: :aws:data-identifier/ -FR InseeCode

arn:aws:dataprotection: :aws:data-identifier/ LatLong

arn:aws:dataprotection::aws:data-identifier/Name

arn:aws:dataprotection: :aws:data-identifier/ -DE NationalIdentificationNumber

arn:aws:dataprotection: :aws:data-identifier/ -ES NationalIdentificationNumber

arn:aws:dataprotection: :aws:data-identifier/ -IT NationalIdentificationNumber

arn:aws:dataprotection: :aws:data-identifier/ -ES NieNumber

arn:aws:dataprotection: :aws:data-identifier/ -ES NifNumber

arn:aws:dataprotection: :aws:data-identifier/ -CA PassportNumber

arn:aws:dataprotection: :aws:data-identifier/ -DE PassportNumber

arn:aws:dataprotection: :aws:data-identifier/ -ES PassportNumber

arn:aws:dataprotection: :aws:data-identifier/ -FR PassportNumber

arn:aws:proteção de dados: :aws:data-identifier/ -GB PassportNumber

arn:aws:dataprotection: :aws:data-identifier/ -IT PassportNumber

arn:aws:dataprotection: :aws:data-identifier/ -US PassportNumber

arn:aws:dataprotection: :aws:data-identifier/ -CA PermanentResidenceNumber

arn:aws:proteção de dados: :aws:data-identifier/ -BR PhoneNumber

arn:aws:dataprotection: :aws:data-identifier/ -DE PhoneNumber

arn:aws:dataprotection: :aws:data-identifier/ -ES PhoneNumber

## ARNs de identificadores de dados de PII

```
arn:aws:dataprotection: :aws:data-identifier/ -FR PhoneNumber
```

```
arn:aws:proteção de dados: :aws:data-identifier/ -GB PhoneNumber
```

```
arn:aws:dataprotection: :aws:data-identifier/ -IT PhoneNumber
```

```
arn:aws:dataprotection: :aws:data-identifier/ -US PhoneNumber
```

```
arn:aws:dataprotection: :aws:data-identifier/ -CA PostalCode
```

```
arn:aws:proteção de dados: :aws:data-identifier/ -BR RgNumber
```

```
arn:aws:dataprotection: :aws:data-identifier/ -CA SocialInsuranceNumber
```

```
arn:aws:dataprotection::aws:data-identifier/Ssn-ES
```

```
arn:aws:dataprotection::aws:data-identifier/Ssn-US
```

```
arn:aws:dataprotection: :aws:data-identifier/ -DE TaxId
```

```
arn:aws:dataprotection: :aws:data-identifier/ -ES TaxId
```

```
arn:aws:dataprotection: :aws:data-identifier/ -FR TaxId
```

```
arn:aws:proteção de dados: :aws:data-identifier/ -GB TaxId
```

```
arn:aws:dataprotection: :aws:data-identifier/ VehicleIdentificationNumber
```

```
arn:aws:dataprotection: :aws:data-identifier/ -US ZipCode
```

## Usar identificadores de dados personalizados no Amazon SNS

### Tópicos

- [O que são identificadores de dados personalizados?](#)
- [Usar identificadores de dados personalizados na política de proteção de dados](#)
- [Restrições de identificadores de dados personalizados](#)

## O que são identificadores de dados personalizados?

Os identificadores de dados personalizados (CDIs) permitem que você defina suas próprias expressões regulares personalizadas que podem ser usadas na política de proteção de dados. Usando identificadores de dados personalizados, é possível segmentar casos de uso de informações de identificação pessoal (PII) específicos da empresa que os [identificadores de dados gerenciados](#) não podem fornecer. Por exemplo, você pode usar um identificador de dados personalizado para procurar IDs de funcionários específicos da empresa. Identificadores de dados personalizados podem ser usados em conjunto com identificadores de dados gerenciados.

## Usar identificadores de dados personalizados na política de proteção de dados

A política de proteção de dados a seguir instrui o tópico do Amazon SNS a detectar cargas úteis que contêm IDs de funcionários específicos da empresa e, depois, mascarar esses IDs usando o símbolo de hash (#).

1. Crie um bloco `Configuration` na política de proteção de dados.
2. Insira um `Name` para o identificador de dados personalizado. Por exemplo, **EmployeeId**.
3. Insira um `Regex` para o identificador de dados personalizado. Por exemplo, **EID-\d{9}-US**.
4. Consulte o identificador de dados personalizado a seguir em uma declaração de política.

```
{
  "Name": "__example_data_protection_policy",
  "Description": "Example data protection policy",
  "Version": "2021-06-01",
  "Configuration": {
    "CustomDataIdentifier": [
      {"Name": "EmployeeId", "Regex": "EID-\d{9}-US"}
    ]
  },
  "Statement": [
    {
      "DataDirection": "Inbound",
      "Principal": ["*"],
      "DataIdentifier": [
        "EmployeeId"
      ],
      "Operation": {
        "Deidentify": {
          "MaskConfig": {
```

```
        "MaskWithCharacter": "#"  
    }  
  }  
} ]  
}
```

5. (Opcional) Continue adicionando identificadores de dados personalizados ao bloco `Configuration`, conforme necessário. No momento, as políticas de proteção de dados são compatíveis com dez identificadores de dados personalizados, no máximo.

## Restrições de identificadores de dados personalizados

Os identificadores de dados personalizados do Amazon SNS têm as seguintes limitações:

- As políticas de proteção de dados são compatíveis com dez identificadores de dados personalizados, no máximo.
- O tamanho máximo dos nomes de identificadores de dados personalizados é 128 caracteres. Os seguintes caracteres são aceitos:
  - Alfanuméricos: (a – z; A – Z; 0 – 9)
  - Símbolos: (“\_” | “-”)
- O RegEx tem um tamanho máximo de 200 caracteres. Os seguintes caracteres são aceitos:
  - Alfanuméricos: (a – z; A – Z; 0 – 9)
  - Símbolos: (“\_” | “#” | “=” | “@” | “/” | “;” | “,” | “-” | “ ”)
  - Caracteres reservados RegEx: (“^” | “\$” | “?” | “[” | “]” | “{” | “}” | “|” | “\” | “\*” | “+” | “.”)
- Os identificadores de dados personalizados não podem compartilhar o mesmo nome de um identificador de dados gerenciados.
- É necessário especificar identificadores de dados personalizados em cada política de proteção de dados para cada tópico do Amazon SNS.

# Entrega de mensagens do Amazon SNS

Esta seção descreve como funciona a entrega de mensagens.

## Tópicos

- [Entrega de mensagens brutas do Amazon SNS](#)
- [Enviar mensagens do Amazon SNS para uma fila do SQS em uma conta diferente](#)
- [Enviar mensagens do Amazon SNS para uma fila do Amazon SQS ou uma função do AWS Lambda em uma conta diferente](#)
- [Status de entrega de mensagens do Amazon SNS](#)
- [Novas tentativas de entrega de mensagens do Amazon SNS](#)
- [Filas de mensagens mortas \(DLQs\) do Amazon SNS](#)

## Entrega de mensagens brutas do Amazon SNS

Para evitar que os endpoints [Amazon Data Firehose](#), [Amazon SQS](#) e HTTP/S processem a formatação JSON das mensagens, o Amazon SNS permite a entrega de mensagens brutas:

- Quando você ativa a entrega de mensagens brutas para endpoints Amazon Data Firehose ou Amazon SQS, qualquer metadado do Amazon SNS é retirado da mensagem publicada e a mensagem é enviada como está.
- Quando você habilita a entrega de mensagens brutas para endpoints HTTP/S, o cabeçalho HTTP `x-amz-sns-rawdelivery` com o valor definido como `true` é adicionado à mensagem, indicando que a mensagem foi publicada sem formatação JSON.
- Quando você habilita a entrega de mensagens brutas para endpoints HTTP/S, o corpo da mensagem, o IP do cliente e os cabeçalhos necessários são entregues. Quando você especifica atributos de mensagem, ela não é enviada.
- Quando você ativa a entrega de mensagens brutas para endpoints Firehose, o corpo da mensagem é entregue. Quando você especifica atributos de mensagem, ela não é enviada.

Para habilitar a entrega de mensagens brutas usando um AWS SDK, você deve usar a ação da `SetSubscriptionAttribute` API e definir o valor do `RawMessageDelivery` atributo como `true`.

## Habilitar a entrega de mensagens brutas usando o AWS Management Console

1. Faça login no [console do Amazon SNS](#).
2. No painel de navegação, escolha Topics (Tópicos).
3. Na página Tópicos, escolha um tópico inscrito em um endpoint Firehose, Amazon SQS ou HTTP/S.
4. Na **MyTopic** página, na seção Assinatura, escolha uma assinatura e escolha Editar.
5. Na página Edit **EXAMPLE1-23bc-4567-d890-ef12g3hij456 (Editar EXEMPL01-23bc-4567-d890-ef12g3hij456)**, na seção Details (Detalhes), escolha Enable raw message delivery (Habilitar a entrega de mensagens brutas).
6. Escolha Salvar alterações.

## Exemplos de formatos de mensagens

Nos exemplos a seguir, a mesma mensagem é enviada para a mesma fila do Amazon SQS duas vezes. A única diferença é que a entrega de mensagens brutas está desabilitada para a primeira mensagem e habilitada para a segunda.

- Entrega de mensagens brutas desabilitada

```
{
  "Type": "Notification",
  "MessageId": "dc1e94d9-56c5-5e96-808d-cc7f68faa162",
  "TopicArn": "arn:aws:sns:us-east-2:111122223333:ExampleTopic1",
  "Subject": "TestSubject",
  "Message": "This is a test message.",
  "Timestamp": "2021-02-16T21:41:19.978Z",
  "SignatureVersion": "1",
  "Signature":
    "FMG5t1ZhJNHLHUXvZgtZz1k24FzVa7oX0T4P03neeXw8ZEXZx6z35j2F0TuNYShn2h0bKNC/
    zLTnMyIxEzmi2X1sh0BWsJHkrW2xkR58ABZF+4uWHEE73yDVR4SyYAikP9jstZzDRm
    +bcVs8+T0yaLiEGLrIIIL4esi11lhIkgErCuy5btPcWXBdio2fpCRD5x9oR6gmE/
    rd5071X1c1uvnv4r1Lkk4pqP2/iUfxFZva1xLSRvgyfm6D9hNk1VyPfy
    +7Ta1MD0lzmJu0rExtnSIbZew3foxgx8GT+1bZkLd0ZdtdRj1IyPRP44eyq78sU0Eo/
    LsDr0Iak4ZDpg8dXg==",
  "SigningCertURL": "https://sns.us-east-2.amazonaws.com/
  SimpleNotificationService-010a507c1833636cd94bdb98bd93083a.pem",
```

```
"UnsubscribeURL": "https://sns.us-east-2.amazonaws.com/?
Action=Unsubscribe&SubscriptionArn=arn:aws:sns:us-
east-2:111122223333:ExampleTopic1:e1039402-24e7-40a3-a0d4-797da162b297"
}
```

- Entrega de mensagens brutas habilitada

```
This is a test message.
```

## Atributos de mensagem e entrega de mensagens brutas para assinaturas do Amazon SQS

O Amazon SNS suporta a entrega de atributos de mensagem, que permitem que você forneça itens de metadados estruturados, como timestamps, dados geoespaciais, assinaturas e identificadores, sobre a mensagem. Para assinaturas do Amazon SQS com entrega de mensagens brutas habilitada, um máximo de 10 atributos de mensagem podem ser enviados. Para enviar mais de 10 atributos de mensagem, você deve desativar a entrega de mensagens brutas. No entanto, o Amazon SNS descarta mensagens com mais de 10 atributos de mensagem direcionadas às assinaturas do Amazon SQS com a entrega de mensagens brutas ativada, tratando-as como erros do lado do cliente.

## Enviar mensagens do Amazon SNS para uma fila do SQS em uma conta diferente

Este documento descreve como publicar uma notificação em um tópico do Amazon SNS com uma ou mais inscrições em filas do Amazon SQS em outra conta. Você pode configurar o tópico e as filas da mesma forma que faria se eles estivessem na mesma conta (consulte [Fanout para filas do Amazon SQS](#)). A maior diferença é a forma de lidar com a confirmação de inscrição, e isso depende de como você inscreve a fila no tópico.

É prática recomendada seguir as etapas mencionadas na seção [Proprietário da fila cria a assinatura](#) quando possível, porque a confirmação é automática quando o proprietário da fila cria a inscrição.

### Note

Se a fila do Amazon SQS tiver um grande volume de mensagens, recomendamos que o proprietário da fila crie a assinatura.

## Tópicos

- [Proprietário da fila cria a assinatura](#)
- [Um usuário que não é proprietário da fila cria uma assinatura](#)
- [Como faço para forçar uma assinatura a exigir autenticação em solicitações de cancelamento de assinatura?](#)

## Proprietário da fila cria a assinatura

A conta que criou a fila do Amazon SQS é proprietária da fila. Quando o proprietário da fila cria uma assinatura, ela não requer confirmação. A fila começa a receber notificações do tópico assim que a ação `Subscribe` é concluída. Para permitir que o proprietário da fila assine o tópico do proprietário, este último deve conceder à conta do proprietário da fila permissão para chamar a ação `Subscribe` no tópico.

### Etapa 1: Definir a política de tópico usando o AWS Management Console

1. Faça login no [console do Amazon SNS](#).
2. No painel de navegação, escolha Topics (Tópicos).
3. Selecione um tópico e escolha Edit (Editar).
4. Na página Edit **MyTopic** (Editar MyTopic), expanda a seção Access policy (Política de acesso).
5. Insira a seguinte política:

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "111122223333"
      },
      "Action": "sns:Subscribe",
      "Resource": "arn:aws:sns:us-east-2:123456789012:MyTopic"
    }
  ]
}
```

Essa política concede à conta 111122223333 permissão para chamar `sns:Subscribe` em `MyTopic` na conta 123456789012.

Um usuário com as credenciais da conta 111122223333 pode assinar MyTopic. Essa permissão possibilita que o ID da conta delegue a permissão ao usuário ou à função do IAM. Somente os usuários da conta raiz ou administrador terão permissão para chamar `sns:Subscribe`. O usuário ou a função do IAM também precisará ter `sns:subscribe` para permitir a assinatura da respectiva fila.

6. Escolha Save changes (Salvar alterações).

Um usuário com as credenciais da conta 111122223333 pode assinar MyTopic.

## Etapa 2: adicionar uma assinatura de fila do Amazon SQS a um tópico em outra Conta da AWS usando o AWS Management Console

Antes de começar, verifique se você tem os ARNs do tópico e da fila e se [concedeu permissão ao tópico para enviar mensagens à fila](#).

1. Faça login no [console do Amazon SQS](#).
2. No painel de navegação, escolha Queues (Filas).
3. Na lista de filas, escolha a fila que você deseja assinar em um tópico do Amazon SNS.
4. Escolha Subscribe to Amazon SNS topic (Inscrever-se no tópico do Amazon SNS).
5. No menu Specify an Amazon SNS topic available for this queue (Especificar um tópico do Amazon SNS disponível para esta fila), escolha o Amazon SNS topic (Tópico do Amazon SNS) para a fila.
6. Selecione Enter Amazon SNS topic ARN (Inserir o ARN do tópico do Amazon SNS) e insira o Amazon Resource Name (ARN) (Nome do recurso da Amazon (ARN)) do tópico.
7. Escolha Save (Salvar).

### Note

- Para poder se comunicar com o serviço, a fila deve ter permissões para o Amazon SNS.
- Como você é o proprietário da fila, você não precisa confirmar a assinatura.

## Um usuário que não é proprietário da fila cria uma assinatura

Qualquer usuário que criar uma assinatura, mas não for o proprietário da fila, deverá confirmar a assinatura.

Quando você usa a ação `Subscribe`, o Amazon SNS envia uma confirmação de assinatura para a fila. A assinatura é exibida no console do Amazon SNS, com seu ID de assinatura definido como Pending Confirmation (Confirmação pendente).

Para confirmar a inscrição, o usuário com permissão para ler mensagens da fila precisa recuperar o URL de confirmação da inscrição e o proprietário da inscrição precisa confirmá-la usando o URL de confirmação. Até que a inscrição seja confirmada, nenhuma notificação publicada no tópico é enviada para a fila. Para confirmar a assinatura, você pode usar o console do Amazon SNS ou a ação `ReceiveMessage`.

### Note

Antes de inscrever um endpoint no tópico, certifique-se de que a fila possa receber mensagens do tópico definindo a permissão `sqs:SendMessage` para a fila. Para obter mais informações, consulte [Etapa 2: Conceder permissão ao tópico do Amazon SNS para enviar mensagens à fila do Amazon SQS](#).

## Etapa 2: adicionar uma assinatura de fila do Amazon SQS a um tópico em outra Conta da AWS usando o AWS Management Console

Antes de começar, verifique se você tem os ARNs do tópico e da fila e se [concedeu permissão ao tópico para enviar mensagens à fila](#).

1. Faça login no [console do Amazon SNS](#).
2. No painel de navegação, escolha Subscriptions (Assinaturas).
3. Na página Subscriptions (Assinaturas), escolha Create subscription (Criar assinatura).
4. Na página Create subscription (Criar inscrição), na seção Details (detalhes), faça o seguinte:
  - a. Para Topic ARN (ARN de tópico), insira o ARN do tópico.
  - b. Em Protocol (Protocolo), escolha Amazon SQS.
  - c. Em Endpoint, insira o ARN de uma fila.
  - d. Selecione Create subscription.

**Note**

- Para poder se comunicar com o serviço, a fila deve ter permissões para o Amazon SNS.

Veja a seguir um exemplo de instrução de política que permite ao tópico do Amazon SNS enviar mensagem à fila do Amazon SQS.

```
{
  "Sid": "Stmt1234",
  "Effect": "Allow",
  "Principal": "*",
  "Action": "sqs:SendMessage",
  "Resource": "arn:aws:sqs:us-west-2:111111111111:QueueName",
  "Condition": {
    "ArnEquals": {
      "aws:SourceArn": "arn:aws:sns:us-west-2:555555555555:TopicName"
    }
  }
}
```

## Etapa 2: Como confirmar uma assinatura com o uso do AWS Management Console

1. Faça login no [console do Amazon SQS](#).
2. Selecione a fila que tem uma inscrição pendente no tópico.
3. Escolha Send and receive messages (Enviar e receber mensagens), depois escolha Poll for messages (Pesquisar mensagens).

Uma mensagem com a confirmação de assinatura é recebida na fila.

4. Na coluna Body (Corpo), faça o seguinte:
  - a. Escolha More Details (Mais detalhes).
  - b. Na caixa de diálogo Message Details (Detalhes da mensagem), localize e anote o valor SubscribeURL. Este é seu link de assinatura (exemplo abaixo). Para obter detalhes adicionais sobre a validação de token de API, consulte [ConfirmSubscription](#) na Referência da API do Amazon SNS.

```
https://sns.us-west-2.amazonaws.com/?
Action=ConfirmSubscription&TopicArn=arn:aws:sns:us-
east-2:123456789012:MyTopic&Token=2336412f37fb...
```

- c. Anote o link de confirmação da inscrição. O URL deve ser transmitido do proprietário da fila para o proprietário da assinatura. O proprietário da assinatura precisa inserir o URL no [console do Amazon SNS](#).
5. Faça login como o proprietário da assinatura no [console do Amazon SNS](#). O proprietário da assinatura executa a confirmação.
6. Escolha o tópico pertinente.
7. Escolha a inscrição pertinente na tabela de listas de assinatura do tópico. Ela é identificada como “Confirmação pendente”.
8. Clique em Confirm subscription (Confirmar assinatura).
9. Uma janela modal é exibida solicitando o link de confirmação da assinatura. Cole o link de confirmação da inscrição.
10. Selecione Confirm subscription (Confirmar assinatura) na janela modal.

Uma resposta em XML é exibida, por exemplo:

```
<ConfirmSubscriptionResponse>
  <ConfirmSubscriptionResult>
    <SubscriptionArn>arn:aws:sns:us-east-2:123456789012:MyTopic:1234a567-
bc89-012d-3e45-6fg7h890123i</SubscriptionArn>
  </ConfirmSubscriptionResult>
  <ResponseMetadata>
    <RequestId>abcd1efg-23hi-jkl4-m5no-p67q8rstuvw9</RequestId>
  </ResponseMetadata>
</ConfirmSubscriptionResponse>
```

A fila inscrita está pronta para receber mensagens do tópico.

11. (Opcional) Se você exibir a assinatura do tópico no console do Amazon SNS, poderá ver que a mensagem Pending Confirmation (Confirmação pendente) foi substituída pelo ARN da assinatura na coluna Subscription ID (ID da assinatura).

## Como faço para forçar uma assinatura a exigir autenticação em solicitações de cancelamento de assinatura?

O proprietário da assinatura precisa definir o sinalizador `AuthenticateOnUnsubscribe` como `true` na confirmação de assinatura.

- `AuthenticateOnUnsubscribe` é definido automaticamente como `true` quando o proprietário da fila cria a inscrição.
- `AuthenticateOnUnsubscribe` não pode ser definido como `true` quando o link de confirmação da assinatura é acessado sem autenticação.

## Enviar mensagens do Amazon SNS para uma fila do Amazon SQS ou uma função do AWS Lambda em uma conta diferente

O Amazon SNS é compatível com entregas entre regiões, tanto para regiões habilitadas por padrão quanto para [regiões de adesão](#). Para obter a lista atual de regiões da AWS compatíveis com o Amazon SNS, incluindo regiões de adesão, consulte [Endpoints e cotas do Amazon Simple Notification Service](#) na Referência geral da Amazon Web Services.

O Amazon SNS oferece suporte à entrega entre regiões de notificações para filas do Amazon SQS e funções do AWS Lambda. Quando uma das regiões é uma região de adesão, você deve especificar um principal de serviço do Amazon SNS diferente na política do recurso inscrito.

O comando de assinatura do Amazon SNS deve ser executado na conta de destino na região em que o Amazon SNS está hospedado. Por exemplo, se o Amazon SNS estiver na conta “A” na região `us-east-1` e a função do Lambda estiver na conta “B” na região `us-east-2`, o comando da CLI da assinatura deverá ser executado na conta “A” na região `us-east-1`.

### Regiões de adesão

O Amazon SNS é compatível com as seguintes regiões de adesão:

Nome da região	Região
Região África (Cidade do Cabo)	<code>af-south-1</code>
Região Ásia-Pacífico (Hong Kong)	<code>ap-east-1</code>

Nome da região	Região
Ásia-Pacífico (Haiderabade)	ap-south-2
Região Ásia-Pacífico (Jacarta)	ap-southeast-3
Região Ásia-Pacífico (Melbourne)	ap-southeast-4
Região Ásia-Pacífico (Osaka)	ap-northeast-3
Região Europa (Milão)	eu-south-1
Região Europa (Espanha)	eu-south-2
Região Europa (Zurique)	eu-central-2
Região de Israel (Tel Aviv)	il-central-1
Região Oriente Médio (Bahrein)	me-south-1
Região do Oriente Médio (Emirados Árabes Unidos)	me-central-1

Para obter informações sobre como habilitar uma região opcional, consulte [Gerenciando AWS regiões](#) no Referência geral da Amazon Web Services.

Quando você usa o Amazon SNS para entregar mensagens de regiões de adesão para regiões habilitadas por padrão, é necessário alterar a política de recurso criada para a fila. Substitua o principal `sns.amazonaws.com` por `sns.<opt-in-region>.amazonaws.com`. Por exemplo:

- Para inscrever uma fila do Amazon SQS no Leste dos EUA (N. da Virgínia) em um tópico do Amazon SNS na Ásia-Pacífico (Hong Kong), altere a entidade principal na política de fila para `sns.ap-east-1.amazonaws.com`. As regiões de adesão incluem as regiões lançadas após 20 de março de 2019, que abrangem Ásia-Pacífico (Hong Kong), Ásia-Pacífico (Jacarta), Oriente Médio (Bahrein), Europa (Milão) e África (Cidade do Cabo). As regiões lançadas antes de 20 de março de 2019 são habilitadas por padrão.

## Compatibilidade da entrega entre regiões com o Amazon SQS

Tipo de entrega entre regiões	Compatível/não compatível	
Região habilitada por padrão para região de adesão	Compatível com o uso de <code>sns.&lt;opt-in-region&gt;.amazonaws.com</code> na entidade principal do serviço para a fila	
Região de adesão para região habilitada por padrão	Compatível com o uso de <code>sns.&lt;opt-in-region&gt;.amazonaws.com</code> na entidade principal do serviço para a fila	
Região de adesão para região de adesão	Não suportado	

Veja a seguir um exemplo de uma declaração de política de acesso que permite que um tópico do Amazon SNS em uma região opcional (af-south-1) seja entregue em uma fila do Amazon SQS em uma região (us-east-1). `enabled-by-default` Ele contém a configuração da entidade principal regionalizada do serviço que é necessária no caminho `Statement/Principal/Service`.

```
{
  "Version": "2008-10-17",
  "Id": "__default_policy_ID",
  "Statement": [
    {
      "Sid": "allow_sns_arn:aws:sns:af-south-1:111111111111:source_topic_name",
      "Effect": "Allow",
      "Principal": {
        "Service": "sns.af-south-1.amazonaws.com"
      },
      "Action": "SQS:SendMessage",
      "Resource": "arn:aws:sqs:us-east-1:111111111111:destination_queue_name",
      "Condition": {
        "ArnLike": {
          "aws:SourceArn": "arn:aws:sns:af-south-1:111111111111:source_topic_name"
        }
      }
    }
  ]
}
```

```

    }
  },
  ...
]
}

```

- Para inscrever uma AWS Lambda função no Leste dos EUA (Norte da Virgínia) em um tópico do Amazon SNS na Ásia-Pacífico (Hong Kong), altere o principal na política da AWS Lambda função para `sns.ap-east-1.amazonaws.com`. As regiões de adesão incluem as regiões lançadas após 20 de março de 2019, que abrangem Ásia-Pacífico (Hong Kong), Ásia-Pacífico (Jacarta), Oriente Médio (Bahrein), Europa (Milão) e África (Cidade do Cabo). As regiões lançadas antes de 20 de março de 2019 são habilitadas por padrão.

### Suporte de entrega entre regiões para AWS Lambda

Tipo de entrega entre regiões	Compatível/não compatível	
Região habilitada por padrão para região de adesão	Não suportado	
Região de adesão para região habilitada por padrão	Compatível com o uso de <code>sns.&lt;opt-in-region&gt;.amazonaws.com</code> na entidade principal do serviço para a função do Lambda	
Região de adesão para região de adesão	Não suportado	

## Status de entrega de mensagens do Amazon SNS

O Amazon SNS fornece suporte para registrar em log o status de entrega de mensagens de notificação enviadas a tópicos com os seguintes endpoints do Amazon SNS:

- HTTP
- Amazon Data Firehose
- AWS Lambda

- Endpoint da aplicação da plataforma
- Amazon Simple Queue Service

Depois de configurar os atributos do status de entrega da mensagem, as entradas de registro são enviadas ao CloudWatch Logs para mensagens enviadas aos assinantes do tópico. O registro de status de entrega de mensagens proporciona um melhor insight operacional, por exemplo:

- Saber se uma mensagem foi entregue para o endpoint do Amazon SNS.
- Identificar a resposta enviada do endpoint do Amazon SNS ao Amazon SNS.
- Determinar o tempo de permanência da mensagem (o tempo entre o carimbo de data e hora da publicação e antes do envio para um endpoint do Amazon SNS).

Para configurar os atributos do tópico para o status de entrega de mensagens, você pode usar os AWS Management Console kits de desenvolvimento de AWS software (SDKs), a API de consulta ou AWS CloudFormation.

## Tópicos

- [Configurar registro em log do status de entrega usando o AWS Management Console](#)
- [Configurando o registro do status de entrega usando os SDKs AWS](#)
- [AWS Exemplos de SDK para configurar atributos de tópicos](#)
- [Configurar registro em log do status de entrega usando o AWS CloudFormation](#)

## Configurar registro em log do status de entrega usando o AWS Management Console

1. Faça login no [console do Amazon SNS](#).
2. No painel de navegação, escolha Topics (Tópicos).
3. Na página Topics (Tópicos), escolha um tópico e selecione Edit (Editar).
4. Na *MyTopic* página Editar, expanda a seção Registro do status de entrega.
5. Escolha o protocolo para os quais você deseja registrar em log o status de entrega, por exemplo, AWS Lambda.
6. Insira a taxa de amostragem de sucesso (a porcentagem de mensagens bem-sucedidas das quais você deseja receber CloudWatch registros).

## 7. Na seção IAM roles (Funções do IAM), siga um destes procedimentos:

- Para escolher uma função de serviço existente da sua conta, escolha Use existing service role (Usar função de serviço existente) e especifique as funções do IAM para entregas bem-sucedidas e com falha.
- Para criar uma nova função de serviço na sua conta, escolha Create new service role (Criar nova função de serviço), selecione Create new roles (Criar novas funções) para definir as funções do IAM para entregas bem-sucedidas e com falha no console do IAM.

Para dar ao Amazon SNS acesso de gravação para usar CloudWatch Logs em seu nome, escolha Permitir.

## 8. Escolha Salvar alterações.

Agora você pode visualizar e analisar os CloudWatch registros que contêm o status de entrega da mensagem. Para obter mais informações sobre o uso CloudWatch, consulte a [CloudWatch documentação](#).

## Configurando o registro do status de entrega usando os SDKs AWS

Os AWS SDKs fornecem APIs em vários idiomas para usar atributos de status de entrega de mensagens com o Amazon SNS.

### Atributos de tópicos

Você pode usar os seguintes valores de nome de atributo de tópico para o status de entrega de mensagens:

#### HTTP

- `HTTPSuccessFeedbackRoleArn`: indica o status de entrega de mensagem bem-sucedida para um tópico do Amazon SNS que está inscrito em um endpoint HTTP.
- `HTTPSuccessFeedbackSampleRate`: indica a porcentagem de mensagens bem-sucedidas para amostrar para um tópico do Amazon SNS que está inscrito em um endpoint HTTP.
- `HTTPFailureFeedbackRoleArn`: indica o status de entrega de mensagem com falha para um tópico do Amazon SNS que está inscrito em um endpoint HTTP.

### Amazon Data Firehose

- `FirehoseSuccessFeedbackRoleArn`: indica o status de entrega de mensagem bem-sucedida para um tópico do Amazon SNS que está inscrito em um endpoint do Amazon Kinesis Data Firehose.
- `FirehoseSuccessFeedbackSampleRate`: indica a porcentagem de mensagens bem-sucedidas para amostrar para um tópico do Amazon SNS que está inscrito em um endpoint do Amazon Kinesis Data Firehose.
- `FirehoseFailureFeedbackRoleArn`: indica o status de entrega de mensagem com falha para um tópico do Amazon SNS que está inscrito em um endpoint do Amazon Kinesis Data Firehose.

## AWS Lambda

- `LambdaSuccessFeedbackRoleArn`: indica o status de entrega de mensagem bem-sucedida para um tópico do Amazon SNS que está inscrito em um endpoint do Lambda.
- `LambdaSuccessFeedbackSampleRate`: indica a porcentagem de mensagens bem-sucedidas para amostrar para um tópico do Amazon SNS que está inscrito em um endpoint do Lambda.
- `LambdaFailureFeedbackRoleArn`: indica o status de entrega de mensagem com falha para um tópico do Amazon SNS que está inscrito em um endpoint do Lambda.

## Endpoint da aplicação da plataforma

- `ApplicationSuccessFeedbackRoleArn`— Indica o status de entrega bem-sucedida de mensagens para um tópico do Amazon SNS que está inscrito em um AWS endpoint do aplicativo.
- `ApplicationSuccessFeedbackSampleRate`— Indica a porcentagem de mensagens bem-sucedidas para amostra de um tópico do Amazon SNS que está inscrito em um AWS endpoint de aplicativo.
- `ApplicationFailureFeedbackRoleArn`— Indica o status de falha na entrega de mensagens para um tópico do Amazon SNS que está inscrito em um AWS endpoint do aplicativo.

### Note

Além de poder configurar os atributos de tópico para o status de entrega de mensagens de notificação enviadas para endpoints da aplicação do Amazon SNS, também é possível configurar os atributos do aplicativo para o status de entrega de mensagens de notificação por push enviadas para serviços de notificação por push. Para obter mais informações,

consulte [Usar atributos de aplicativo do Amazon SNS para obter o status de entrega de mensagens](#).

## Amazon SQS

- `SQSSuccessFeedbackRoleArn`: indica o status de entrega de mensagem bem-sucedida para um tópico do Amazon SNS que está inscrito em um endpoint do Amazon SQS.
- `SQSSuccessFeedbackSampleRate`: indica a porcentagem de mensagens bem-sucedidas para amostrar para um tópico do Amazon SNS que está inscrito em um endpoint do Amazon SQS.
- `SQSFailureFeedbackRoleArn`: indica o status de entrega de mensagem com falha para um tópico do Amazon SNS que está inscrito em um endpoint do Amazon SQS.

### Note

Os `<ENDPOINT>FailureFeedbackRoleArn` atributos `<ENDPOINT>SuccessFeedbackRoleArn` e são usados para dar ao Amazon SNS acesso de gravação para usar CloudWatch Logs em seu nome. O atributo `<ENDPOINT>SuccessFeedbackSampleRate` é para especificar a porcentagem de taxa de amostra (0-100) de mensagens bem-sucedidas. Depois de configurar o `<ENDPOINT>FailureFeedbackRoleArn` atributo, todas as entregas de mensagens com falha geram CloudWatch registros.

## AWS Exemplos de SDK para configurar atributos de tópicos

Os exemplos de código a seguir mostram como usar `SetTopicAttributes`.

### CLI

#### AWS CLI

Para definir um atributo para um tópico

O exemplo `set-topic-attributes` a seguir define o atributo `DisplayName` para o tópico especificado.

```
aws sns set-topic-attributes \
```

```
--topic-arn arn:aws:sns:us-west-2:123456789012:MyTopic \  
--attribute-name DisplayName \  
--attribute-value MyTopicDisplayName
```

Este comando não produz saída.

- Para obter detalhes da API, consulte [SetTopicAttributes](#) em Referência de AWS CLI Comandos.

## Java

### SDK para Java 2.x

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```
import software.amazon.awssdk.regions.Region;  
import software.amazon.awssdk.services.sns.SnsClient;  
import software.amazon.awssdk.services.sns.model.SetTopicAttributesRequest;  
import software.amazon.awssdk.services.sns.model.SetTopicAttributesResponse;  
import software.amazon.awssdk.services.sns.model.SnsException;  
  
/**  
 * Before running this Java V2 code example, set up your development  
 * environment, including your credentials.  
 *  
 * For more information, see the following documentation topic:  
 *  
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-  
started.html  
 */  
public class SetTopicAttributes {  
  
    public static void main(String[] args) {  
        final String usage = ""  
  
            Usage:    <attribute> <topicArn> <value>  
  
            Where:
```

```
        attribute - The attribute action to use. Valid parameters are:
Policy | DisplayName | DeliveryPolicy .
        topicArn - The ARN of the topic.\s
        value - The value for the attribute.
        """;

    if (args.length < 3) {
        System.out.println(usage);
        System.exit(1);
    }

    String attribute = args[0];
    String topicArn = args[1];
    String value = args[2];

    SnsClient snsClient = SnsClient.builder()
        .region(Region.US_EAST_1)
        .build();

    setTopAttr(snsClient, attribute, topicArn, value);
    snsClient.close();
}

public static void setTopAttr(SnsClient snsClient, String attribute, String
topicArn, String value) {
    try {
        SetTopicAttributesRequest request =
SetTopicAttributesRequest.builder()
            .attributeName(attribute)
            .attributeValue(value)
            .topicArn(topicArn)
            .build();

        SetTopicAttributesResponse result =
snsClient.setTopicAttributes(request);
        System.out.println(
            "\n\nStatus was " + result.sdkHttpResponse().statusCode() +
"\n\nTopic " + request.topicArn()
                + " updated " + request.attributeName() + " to " +
request.attributeValue());
    } catch (SnsException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}
```

```
    }  
  }  
}
```

- Para obter detalhes da API, consulte [SetTopicAttributes](#) na Referência AWS SDK for Java 2.x da API.

## JavaScript

### SDK para JavaScript (v3)

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

Crie o cliente em um módulo separado e exporte-o.

```
import { SNSClient } from "@aws-sdk/client-sns";  
  
// The AWS Region can be provided here using the `region` property. If you leave  
// it blank  
// the SDK will default to the region set in your AWS config.  
export const snsClient = new SNSClient({});
```

Importe o SDK e os módulos do cliente e chame a API.

```
import { SetTopicAttributesCommand } from "@aws-sdk/client-sns";  
import { snsClient } from "../libs/snsClient.js";  
  
export const setTopicAttributes = async (  
  topicArn = "TOPIC_ARN",  
  attributeName = "DisplayName",  
  attributeValue = "Test Topic",  
) => {  
  const response = await snsClient.send(  
    new SetTopicAttributesCommand({  
      AttributeName: attributeName,
```

```
        AttributeValue: attributeValue,
        TopicArn: topicArn,
    })),
);
console.log(response);
// {
//   '$metadata': {
//     httpStatusCode: 200,
//     requestId: 'd1b08d0e-e9a4-54c3-b8b1-d03238d2b935',
//     extendedRequestId: undefined,
//     cfId: undefined,
//     attempts: 1,
//     totalRetryDelay: 0
//   }
// }
return response;
};
```

- Para obter mais informações, consulte o [Guia do desenvolvedor do AWS SDK for JavaScript](#).
- Para obter detalhes da API, consulte [SetTopicAttributes](#) na Referência AWS SDK for JavaScript da API.

## Kotlin

### SDK for Kotlin

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```
suspend fun setTopAttr(attribute: String?, topicArnVal: String?, value: String?)
{
    val request = SetTopicAttributesRequest {
        attributeName = attribute
        attributeValue = value
        topicArn = topicArnVal
    }
```

```
    }

    SnsClient { region = "us-east-1" }.use { snsClient ->
        snsClient.setTopicAttributes(request)
        println("Topic ${request.topicArn} was updated.")
    }
}
```

- Para obter detalhes da API, consulte a [SetTopicAttributes](#) referência da API AWS SDK for Kotlin.

## PHP

### SDK para PHP

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```
require 'vendor/autoload.php';

use Aws\Exception\AwsException;
use Aws\Sns\SnsClient;

/**
 * Configure the message delivery status attributes for an Amazon SNS Topic.
 *
 * This code expects that you have AWS credentials set up per:
 * https://docs.aws.amazon.com/sdk-for-php/v3/developer-guide/
 * guide_credentials.html
 */

$SnSclient = new SnsClient([
    'profile' => 'default',
    'region' => 'us-east-1',
    'version' => '2010-03-31'
]);
```

```
$attribute = 'Policy | DisplayName | DeliveryPolicy';
$value = 'First Topic';
$topic = 'arn:aws:sns:us-east-1:111122223333:MyTopic';

try {
    $result = $SnSClient->setTopicAttributes([
        'AttributeName' => $attribute,
        'AttributeValue' => $value,
        'TopicArn' => $topic,
    ]);
    var_dump($result);
} catch (AwsException $e) {
    // output error message if fails
    error_log($e->getMessage());
}
```

- Para obter detalhes da API, consulte [SetTopicAttributes](#) a Referência AWS SDK for PHP da API.

## Ruby

### SDK for Ruby

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
# Service class to enable an SNS resource with a specified policy
class SnsResourceEnabler
  # Initializes the SnsResourceEnabler with an SNS resource client
  #
  # @param sns_resource [Aws::SNS::Resource] The SNS resource client
  def initialize(sns_resource)
    @sns_resource = sns_resource
    @logger = Logger.new($stdout)
  end
end
```

```
# Sets a policy on a specified SNS topic
#
# @param topic_arn [String] The ARN of the SNS topic
# @param resource_arn [String] The ARN of the resource to include in the policy
# @param policy_name [String] The name of the policy attribute to set
def enable_resource(topic_arn, resource_arn, policy_name)
  policy = generate_policy(topic_arn, resource_arn)
  topic = @sns_resource.topic(topic_arn)

  topic.set_attributes({
    attribute_name: policy_name,
    attribute_value: policy
  })

  @logger.info("Policy #{policy_name} set successfully for topic
#{topic_arn}.")
  rescue Aws::SNS::Errors::ServiceError => e
    @logger.error("Failed to set policy: #{e.message}")
  end

private

# Generates a policy string with dynamic resource ARNs
#
# @param topic_arn [String] The ARN of the SNS topic
# @param resource_arn [String] The ARN of the resource
# @return [String] The policy as a JSON string
def generate_policy(topic_arn, resource_arn)
  {
    Version: "2008-10-17",
    Id: "__default_policy_ID",
    Statement: [{
      Sid: "__default_statement_ID",
      Effect: "Allow",
      Principal: { "AWS": "*" },
      Action: ["SNS:Publish"],
      Resource: topic_arn,
      Condition: {
        ArnEquals: {
          "AWS:SourceArn": resource_arn
        }
      }
    }]
  }.to_json
end
```

```
end

# Example usage:
if $PROGRAM_NAME == __FILE__
  topic_arn = "MY_TOPIC_ARN"      # Should be replaced with a real topic ARN
  resource_arn = "MY_RESOURCE_ARN" # Should be replaced with a real resource ARN
  policy_name = "POLICY_NAME"     # Typically, this is "Policy"

  sns_resource = Aws::SNS::Resource.new
  enabler = SnsResourceEnabler.new(sns_resource)

  enabler.enable_resource(topic_arn, resource_arn, policy_name)
end
```

- Para obter mais informações, consulte o [Guia do desenvolvedor do AWS SDK for Ruby](#).
- Para obter detalhes da API, consulte [SetTopicAttributes](#) a Referência AWS SDK for Ruby da API.

## SAP ABAP

### SDK para SAP ABAP

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```
TRY.
  lo_sns->settopicattributes(
    iv_topicarn = iv_topic_arn
    iv_attributename = iv_attribute_name
    iv_attributevalue = iv_attribute_value
  ).
  MESSAGE 'Set/updated SNS topic attributes.' TYPE 'I'.
CATCH /aws1/cx_snsnotfoundexception.
  MESSAGE 'Topic does not exist.' TYPE 'E'.
ENDTRY.
```

- Para obter detalhes da API, consulte a [SetTopicAttributes](#) referência da API AWS SDK for SAP ABAP.

## Configurar registro em log do status de entrega usando o AWS CloudFormation

Para configurar `DeliveryStatusLogging` usando AWS CloudFormation, use um modelo JSON ou YAML para criar uma AWS CloudFormation pilha. Para obter mais informações, consulte a `DeliveryStatusLogging` propriedade do `AWS::SNS::Topic` recurso no Guia do AWS CloudFormation usuário. Abaixo estão exemplos de AWS CloudFormation modelos em JSON e YAML para criar um novo tópico ou atualizar um tópico existente com todos os `DeliveryStatusLogging` atributos do protocolo Amazon SQS.

### JSON

```
"Resources": {
  "MySNSTopic" : {
    "Type" : "AWS::SNS::Topic",
    "Properties" : {
      "TopicName" : "TestTopic",
      "DisplayName" : "TEST",
      "SignatureVersion" : "2",
      "DeliveryStatusLogging" : [{
        "Protocol": "sqs",
        "SuccessFeedbackSampleRate": "45",
        "SuccessFeedbackRoleArn": "arn:aws:iam::123456789012:role/
SNSSuccessFeedback_test1",
        "FailureFeedbackRoleArn": "arn:aws:iam::123456789012:role/
SNSFailureFeedback_test2"
      }]
    }
  }
}
```

### YAML

```
Resources:
  MySNSTopic:
    Type: AWS::SNS::Topic
```

```
Properties:
  TopicName:TestTopic
  DisplayName:TEST
  SignatureVersion:2
  DeliveryStatusLogging:
    - Protocol: sqs
      SuccessFeedbackSampleRate: 45
      SuccessFeedbackRoleArn: arn:aws:iam::123456789012:role/
SNSSuccessFeedback_test1
      FailureFeedbackRoleArn: arn:aws:iam::123456789012:role/
SNSFailureFeedback_test2
```

## Novas tentativas de entrega de mensagens do Amazon SNS

O Amazon SNS define uma política de entrega para cada protocolo de entrega. A política de entrega define como o Amazon SNS tenta novamente a entrega de mensagens quando ocorrem erros do lado do servidor (quando o sistema que hospeda o endpoint inscrito se torna indisponível). Quando a política de entrega estiver esgotada, o Amazon SNS parará de tentar novamente a entrega e descartará a mensagem, a menos que uma fila de mensagens mortas esteja anexada à inscrição. Para ter mais informações, consulte [Filas de mensagens mortas \(DLQs\) do Amazon SNS](#).

### Tópicos

- [Protocolos e políticas de entrega](#)
- [Estágios da política de entrega](#)
- [Como criar uma política de entrega HTTP/S](#)

## Protocolos e políticas de entrega

### Note

- Com exceção do HTTP/S, não é possível alterar as políticas de entrega definidas pelo Amazon SNS. Somente HTTP/S oferece suporte a políticas personalizadas. Consulte [Como criar uma política de entrega HTTP/S](#).
- O Amazon SNS aplica instabilidade às novas tentativas de entrega. Para obter mais informações, consulte a publicação [Exponential Backoff and Jitter](#) (“Recuo exponencial e instabilidade”) no Blog de arquitetura da AWS .

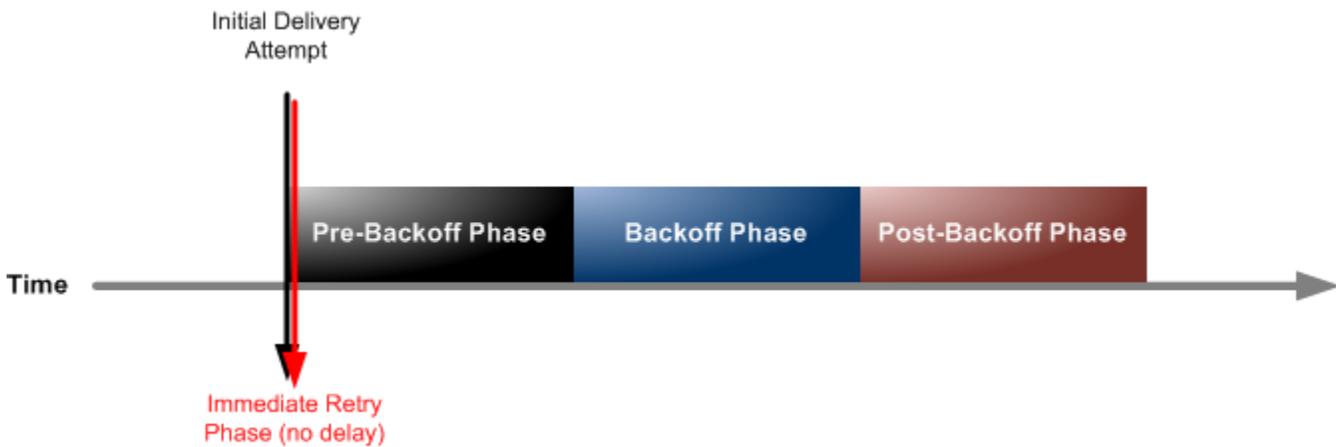
- O tempo total de repetição da política para um endpoint HTTP/S não pode ser maior que 3.600 segundos. Esse limite é fixo e não pode ser alterado.

Tipo de endpoint	Protocolos de entrega	Fase de nova tentativa imediata (sem atraso)	Fase de pré-recuo	Fase de recuo	Fase de pós-recuo	Total de tentativas
AWS endpoints gerenciados	Amazon Data Firehose <sup>1</sup>	3 vezes, sem atraso	2 vezes, 1 segundo de intervalo	10 vezes, com recuo exponencial, de 1 segundo a 20 segundos	100.000 vezes, 20 segundos de intervalo	100.015 vezes, ao longo de 23 dias
	AWS Lambda					
	Amazon SQS					
Endpoints gerenciados pelo cliente	SMTP	0 vezes, sem atraso	2 vezes, 10 segundos de intervalo	10 vezes, com recuo exponencial, de 10 segundos a 600 segundos (10 minutos)	38 vezes, 600 segundos (10 minutos) de intervalo	50 tentativas, mais de 6 horas
	SMS					
	Push para dispositivos móveis					

<sup>1</sup> Para erros de limitação com o protocolo Firehose, o Amazon SNS usa a mesma política de entrega dos endpoints gerenciados pelo cliente.

## Estágios da política de entrega

O diagrama a seguir mostra as fases de uma política de entrega.



Cada política de entrega é composta por quatro fases.

1. Fase de nova tentativa imediata (sem atraso): essa fase ocorre imediatamente após a tentativa inicial de entrega. Não há um intervalo entre novas tentativas nessa fase.
2. Fase de pré-recuo: segue a fase de nova tentativa imediata. Essa fase é usada pelo Amazon SNS para executar um conjunto de novas tentativas antes da aplicação de uma função de recuo. Essa fase especifica o número de novas tentativas e a quantidade de atraso entre elas.
3. Fase de recuo: essa fase controla o atraso entre as novas tentativas usando a função de recuo de nova tentativa. Essa fase define um atraso mínimo, um atraso máximo e uma função de recuo de nova tentativa que define a rapidez com que o atraso aumenta do atraso mínimo para o máximo. A função de recuo pode ser aritmética, exponencial, geométrica ou linear.
4. Fase de pós-recuo: essa fase ocorre após a fase de recuo. Ela especifica um número de novas tentativas e a quantidade de atraso entre elas. Esta é a fase final.

## Como criar uma política de entrega HTTP/S

Use uma política de entrega e as quatro fases para definir como o Amazon SNS executa novas tentativas de entrega de mensagens para endpoints HTTP/S. Com o Amazon SNS, será possível substituir a política de novas tentativas padrão por endpoints HTTP se você, por exemplo, quiser personalizar a política com base na capacidade do servidor HTTP.

É possível definir sua política de entrega HTTP/S como um objeto JSON no nível da inscrição ou do tópico. Ao definir a política no nível do tópico, ela se aplicará a todas as inscrições HTTP/S associadas ao tópico. Para definir a política de entrega no nível da assinatura, você pode usar a ação de API [Subscribe](#) ou [SetSubscriptionAttributes](#). Para definir a política de entrega no nível do tópico, você pode usar a ação de API [CreateTopic](#) ou [SetTopicAttributes](#).

Como alternativa, você também pode usar o [AWS::SNS::Subscription](#) recurso em seus AWS CloudFormation modelos.

Você deve personalizar sua política de entrega de acordo com a capacidade do seu servidor HTTP/S. É possível definir a política como um atributo de tópico ou um atributo de inscrição. Se todas as inscrições HTTP/S no tópico tiverem como destino o mesmo servidor HTTP/S, recomendamos que você defina a política de entrega como um atributo de tópico, para que ela permaneça válida para todas as inscrições HTTP/S no tópico. Caso contrário, você deverá compor uma política de entrega para cada inscrição HTTP/S em seu tópico, de acordo com a capacidade do servidor HTTP/S que a política tem como alvo.

Também é possível definir o cabeçalho Content-Type na política de solicitação para especificar o tipo de mídia da notificação. Por padrão, o Amazon SNS envia todas as notificações para endpoints HTTP/S com o tipo de conteúdo definido como `text/plain; charset=UTF-8`. O Amazon SNS permite que você substitua a política de solicitação padrão. Consulte a tabela a seguir para obter detalhes sobre [headerContentType](#) compatíveis e restrições.

O objeto JSON a seguir representa uma política de entrega que orienta o Amazon SNS a tentar novamente uma entrega HTTP/S que falhou da seguinte forma:

1. 3 vezes imediatamente na fase sem atraso
2. 2 vezes (1 segundo de intervalo) na fase de pré-recuo
3. 10 vezes (com recuo exponencial de 1 segundo a 60 segundos)
4. 35 vezes (com intervalo de 60 segundos) na fase de pós-recuo

Neste exemplo de política de entrega, o Amazon SNS faz ao todo 50 tentativas antes de descartar a mensagem. Para manter a mensagem após as tentativas especificadas na política de entrega serem esgotadas, configure sua assinatura para mover mensagens não entregues para uma fila de mensagens mortas (DLQ). Para ter mais informações, consulte [Filas de mensagens mortas \(DLQs\) do Amazon SNS](#).

#### Note

Essa política de entrega também instrui o Amazon SNS a limitar as entregas a não mais de 10 por segundo, usando a propriedade `maxReceivesPerSecond`. Essa taxa de autolimitação pode resultar em mais mensagens publicadas (tráfego de entrada) do que entregues (tráfego de saída). Se houver mais tráfego de entrada que de saída, sua assinatura pode acumular um grande backlog de mensagens, o que pode causar alta latência

na entrega de mensagens. Em suas políticas de entrega, certifique-se de especificar um valor para `maxReceivesPerSecond` que não afete negativamente o seu workload.

### Note

Essa política de entrega substitui o tipo de conteúdo padrão para notificação HTTP/S por `application/json`.

```
{
  "healthyRetryPolicy": {
    "minDelayTarget": 1,
    "maxDelayTarget": 60,
    "numRetries": 50,
    "numNoDelayRetries": 3,
    "numMinDelayRetries": 2,
    "numMaxDelayRetries": 35,
    "backoffFunction": "exponential"
  },
  "throttlePolicy": {
    "maxReceivesPerSecond": 10
  },
  "requestPolicy": {
    "headerContentType": "application/json"
  }
}
```

A política de entrega é composta por uma política de novas tentativas, uma política de controle de utilização e uma política de solicitação. No total, há nove atributos em uma política de entrega.

Política	Descrição	Restrição
<code>minDelayTarget</code>	O atraso mínimo para uma nova tentativa.  Unidade: segundos	1 a atraso máximo  Padrão: 20
<code>maxDelayTarget</code>	O atraso máximo para uma nova tentativa.	Atraso mínimo a 3.600

Política	Descrição	Restrição
	Unidade: segundos	Padrão: 20
<code>numRetries</code>	O número total de novas tentativas, incluindo repetições imediatas, pré-recuo, recuo e pós-recuo.	0 a 100 Padrão: 3
<code>numNoDelayRetries</code>	O número de novas tentativas a serem feitas imediatamente, sem atraso entre elas.	0 ou mais Padrão: 0
<code>numMinDelayRetries</code>	O número de tentativas na fase de pré-recuo, com o atraso mínimo especificado entre elas.	0 ou mais Padrão: 0
<code>numMaxDelayRetries</code>	O número de novas tentativas na fase pós-recuo, com o atraso máximo entre elas.	0 ou mais Padrão: 0
<code>backoffFunction</code>	O modelo para recuo entre novas tentativas.	Uma das quatro opções: <ul style="list-style-type: none"> <li>• aritmética</li> <li>• exponencial</li> <li>• geométrica</li> <li>• linear</li> </ul> Padrão: linear
<code>maxReceivesPerSecond</code>	O número máximo de entregas por segundo, por inscrição.	1 ou mais Padrão: sem limitação

Política	Descrição	Restrição
headerContentType	O tipo de conteúdo da notificação que está sendo enviada aos endpoints HTTP/S.	<p>Se a política de solicitação não estiver definida, o tipo de conteúdo usará <code>text/plain; charset=UTF-8</code> como padrão.</p> <p>Quando a entrega de mensagens brutas é desativada para uma assinatura (padrão), ou quando a política de entrega é definida no nível do tópico, os tipos de conteúdo de cabeçalho compatíveis são <code>application/json</code> e <code>text/plain</code>.</p> <p>Quando a entrega de mensagens brutas é ativada para uma assinatura, os seguintes tipos de conteúdo são compatíveis:</p> <ul style="list-style-type: none"><li>• <code>text/css</code></li><li>• <code>text/csv</code></li><li>• <code>text/html</code></li><li>• <code>text/plain</code></li><li>• <code>text/xml</code></li><li>• <code>application/atom+xml</code></li><li>• <code>application/json</code></li><li>• <code>application/octet-stream</code></li><li>• <code>application/soap+xml</code></li><li>• <code>aplicação/ x-www-form-urlencoded</code></li></ul>

Política	Descrição	Restrição
		<ul style="list-style-type: none"><li>• application/xhtml+xml</li><li>• application/xml</li></ul>

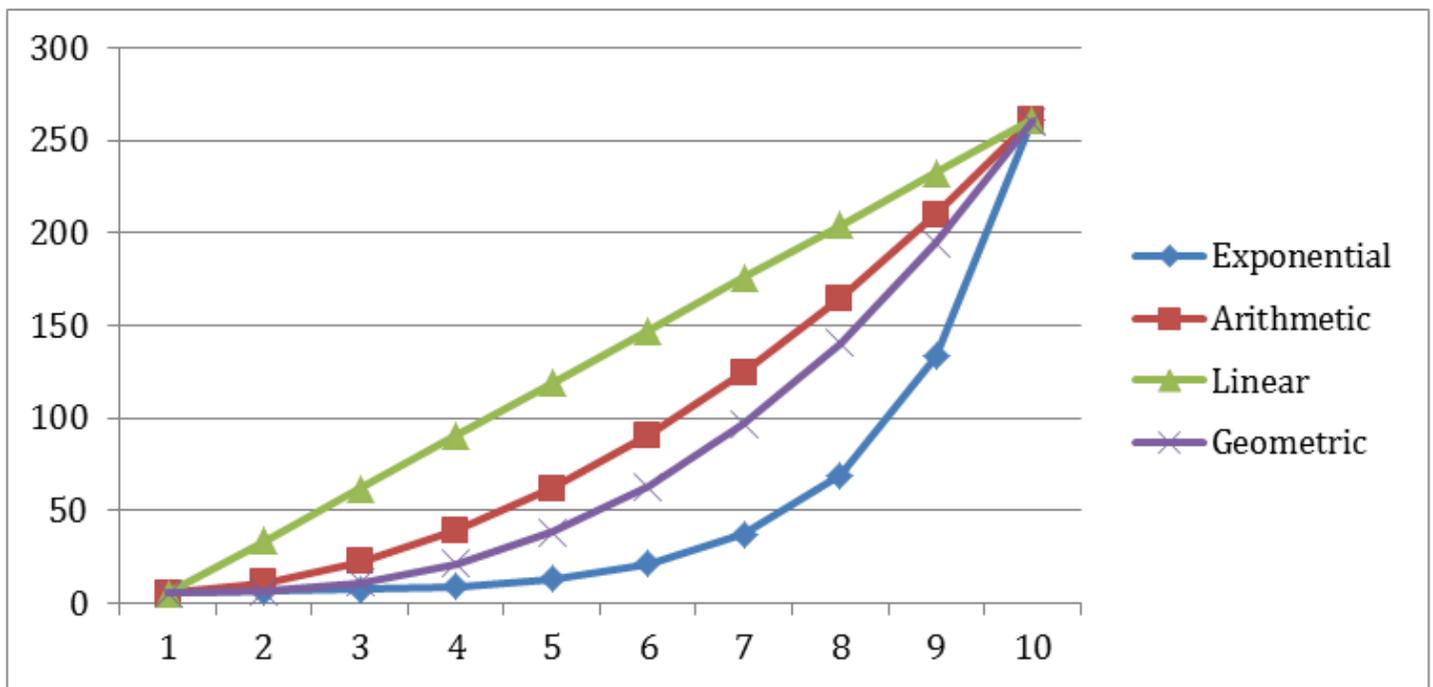
O Amazon SNS usa a seguinte fórmula para calcular o número de novas tentativas na fase de recuo:

```
numRetries - numNoDelayRetries - numMinDelayRetries - numMaxDelayRetries
```

É possível usar três parâmetros para controlar a frequência de tentativas na fase de recuo.

- `minDelayTarget`: define o atraso associado à primeira nova tentativa de na fase de recuo.
- `maxDelayTarget`: define o atraso associado à tentativa final na fase de recuo.
- `backoffFunction`: define o algoritmo que o Amazon SNS usa para calcular os atrasos associados a todas as novas tentativas entre a primeira e a última nova tentativa na fase de recuo. Você pode usar uma das quatro funções de recuo de nova tentativa.

O diagrama a seguir mostra como cada função de tentativa de recuo afeta o atraso associado a novas tentativas durante a fase de recuo: Uma política de entrega com o número total de tentativas definido como 10, o atraso mínimo definido como 5 segundos e o atraso máximo definido como 260 segundos. O eixo vertical representa o atraso em segundos associado a cada uma das 10 novas tentativas. O eixo horizontal representa o número de novas tentativas, da primeira à décima.



## Filas de mensagens mortas (DLQs) do Amazon SNS

Uma fila de mensagens mortas é uma fila do Amazon SQS para a qual uma assinatura do Amazon SNS pode enviar mensagens que não podem ser entregues aos assinantes com êxito. As mensagens que não podem ser entregues devido a erros do cliente ou erros do servidor são mantidas na fila de mensagens mortas para análise ou reprocessamento adicionais. Para obter mais informações, consulte [Configurar uma fila de mensagens mortas do Amazon SNS para uma assinatura](#) e [Novas tentativas de entrega de mensagens do Amazon SNS](#).

### Note

- A assinatura do Amazon SNS e a fila do Amazon SQS devem estar na mesma AWS e região.
- Em um [tópico FIFO](#), você pode usar uma fila do Amazon SQS como uma fila de mensagens não entregues para a assinatura do Amazon SNS. As assinaturas de tópicos FIFO usam filas FIFO e as de tópicos padrão usam filas padrão.
- Para usar uma fila criptografada do Amazon SQS como uma fila de mensagens não entregues, você deve usar um KMS personalizado com uma política de chave que conceda à entidade principal do serviço Amazon SNS acesso a ações da API do AWS KMS. Para obter mais informações, consulte [Criptografia inativa](#) neste guia e [Proteção de](#)

[dados do Amazon SQS usando criptografia no lado do servidor \(SSE\) eAWS KMS](#)noGuia do desenvolvedor do Amazon Simple Queue Service.

## Tópicos

- [Por que há falha nas entregas de mensagens?](#)
- [Como as filas de mensagens mortas funcionam?](#)
- [Como as mensagens são movidas para uma fila de mensagens mortas?](#)
- [Como posso mover mensagens de uma fila de mensagens mortas?](#)
- [Como posso monitorar e registrar em log filas de mensagens mortas?](#)
- [Configurar uma fila de mensagens mortas do Amazon SNS para uma assinatura](#)

## Por que há falha nas entregas de mensagens?

Em geral, há falha na entrega de mensagens quando o Amazon SNS não pode acessar um endpoint inscrito devido a um erro do lado do cliente ou um erro do lado do servidor. Quando o Amazon SNS recebe um erro do lado do cliente ou continua a receber um erro do lado do servidor para uma mensagem além do número de tentativas especificado pela política de novas tentativas correspondente, o Amazon SNS descarta a mensagem, a menos que uma fila de mensagens mortas esteja anexada à inscrição. Entregas com falha não alteram o status de suas inscrições. Para obter mais informações, consulte [Novas tentativas de entrega de mensagens do Amazon SNS](#).

### Erros no lado do cliente,

Erros do lado do cliente podem ocorrer quando o Amazon SNS tem metadados de inscrição obsoletos. Esses erros geralmente ocorrem quando um proprietário exclui o endpoint (por exemplo, uma função do Lambda inscrita em um tópico do Amazon SNS) ou quando um proprietário altera a política anexada ao endpoint inscrito de uma forma que impede que o Amazon SNS entregue as mensagens para o ponto de endpoint. O Amazon SNS não tenta novamente a entrega da mensagem que falha como resultado de um erro do lado do cliente.

### Erros no lado do servidor

Erros do lado do servidor podem ocorrer quando o sistema responsável pelo endpoint inscrito se torna indisponível ou retorna uma exceção que indica que ele não pode processar uma solicitação

válida do Amazon SNS. Quando ocorrem erros do lado do servidor, o Amazon SNS tenta novamente as entregas com falha usando uma função de recuo exponencial ou linear. Para erros do lado do servidor causados por endpoint gerenciados pela AWS com suporte do Amazon SQS ou do AWS Lambda, o Amazon SNS tenta novamente a entrega até 100.015 vezes, ao longo de 23 dias.

Os endpoints gerenciados pelo cliente (como HTTP, SMTP, SMS ou push em dispositivos móveis) também podem causar erros no lado do servidor. O Amazon SNS tenta novamente a entrega para esses tipos de endpoints também. Enquanto os endpoints HTTP oferecem suporte a políticas de novas tentativas definidas pelo cliente, o Amazon SNS define uma política de novas tentativas de entrega interna como 50 vezes ao longo de 6 horas, para SMTP, SMS e endpoints de push para dispositivos móveis.

## Como as filas de mensagens mortas funcionam?

Uma fila de mensagens mortas é anexada a uma inscrição do Amazon SNS (em vez de um tópico) porque as entregas de mensagens acontecem no nível de inscrição. Isso permite identificar o endpoint de destino original para cada mensagem com mais facilidade.

Uma fila de mensagens mortas associada a uma inscrição do Amazon SNS é uma fila comum do Amazon SQS. Para obter mais informações sobre o período de retenção de mensagens, consulte [Quotas Related to Messages](#) (“Cotas relacionadas a mensagens”) no Guia do desenvolvedor do Amazon Simple Queue Service. É possível alterar o período de retenção de mensagens usando a ação da API [SetQueueAttributes](#) do Amazon SQS. Para tornar seus aplicativos mais resilientes, recomendamos definir o período máximo de retenção para filas de mensagens mortas como 14 dias.

## Como as mensagens são movidas para uma fila de mensagens mortas?

As mensagens são movidas para uma fila de mensagens mortas por meio de uma política de redirecionamento. Uma política de redirecionamento é um objeto JSON que se refere ao ARN da fila de mensagens mortas. O atributo `deadLetterTargetArn` especifica o ARN. O ARN deve apontar para uma fila do Amazon SQS na mesma conta e região da Conta da AWS que sua assinatura do Amazon SNS. Para obter mais informações, consulte [Configurar uma fila de mensagens mortas do Amazon SNS para uma assinatura](#).

O objeto JSON a seguir é uma política de redirecionamento de exemplo, anexada a uma assinatura SNS.

```
{
```

```
"deadLetterTargetArn": "arn:aws:sqs:us-east-2:123456789012:MyDeadLetterQueue"  
}
```

## Como posso mover mensagens de uma fila de mensagens mortas?

É possível mover mensagens para fora de uma fila de mensagens mortas de duas maneiras:

- Evitar escrever lógica do consumidor do Amazon SQS: defina sua fila de mensagens mortas como uma fonte de evento para a função do Lambda drenar sua fila de mensagens mortas.
- Escrever lógica do consumidor do Amazon SQS: use a API do Amazon SQS, o AWS SDK ou a AWS CLI para escrever a lógica personalizada do consumidor para sondagem, processamento e exclusão das mensagens na fila de mensagens mortas.

## Como posso monitorar e registrar em log filas de mensagens mortas?

É possível usar métricas do Amazon CloudWatch para monitorar filas de mensagens mortas associadas às suas inscrições do Amazon SNS. Todas as filas do Amazon SQS emitem métricas do CloudWatch em intervalos de um minuto. Para obter mais informações, consulte [Available CloudWatch Metrics for Amazon SQS](#) (“Métricas disponíveis do CloudWatch para Amazon SQS”) no Guia do desenvolvedor do Amazon Simple Queue Service. Todas as inscrições do Amazon SNS com filas de mensagens mortas também emitem métricas do CloudWatch. Para obter mais informações, consulte [Monitorar tópicos do Amazon SNS usando o CloudWatch](#).

Para ser notificado sobre atividades em suas filas de mensagens mortas, use métricas e alarmes do CloudWatch. Por exemplo, quando você espera que a fila de mensagens mortas esteja sempre vazia, é possível criar um alarme do CloudWatch para a métrica `NumberOfMessagesSent`. Defina o limite do alarme como 0 e especifique um tópico do Amazon SNS a ser notificado quando o alarme for disparado. Esse tópico do Amazon SNS pode entregar sua notificação de alarme a qualquer tipo de ponto de endpoint (como um endereço de e-mail, número de telefone ou aplicativo de pager móvel).

É possível usar o CloudWatch Logs para investigar as exceções que fazem com que haja falha em todas as entregas do Amazon SNS e para que as mensagens sejam enviadas para filas de mensagens mortas. O Amazon SNS pode registrar entregas bem-sucedidas e com falha no CloudWatch. Para obter mais informações, consulte [Status de entrega de mensagens do Amazon SNS](#).

# Configurar uma fila de mensagens mortas do Amazon SNS para uma assinatura

Uma fila de mensagens mortas é uma fila do Amazon SQS para a qual uma assinatura do Amazon SNS pode enviar mensagens que não podem ser entregues aos assinantes com êxito. As mensagens que não podem ser entregues devido a erros do cliente ou erros do servidor são mantidas na fila de mensagens mortas para análise ou reprocessamento adicionais. Para ter mais informações, consulte [Filas de mensagens mortas \(DLQs\) do Amazon SNS](#) e [Novas tentativas de entrega de mensagens do Amazon SNS](#).

Esta página mostra como você pode usar o AWS Management Console, um AWS SDK AWS CLI, o e AWS CloudFormation para configurar uma fila de mensagens mortas para uma assinatura do Amazon SNS.

## Note

Em um [tópico FIFO](#), você pode usar uma fila do Amazon SQS como uma fila de mensagens não entregues para a assinatura do Amazon SNS. As assinaturas de tópicos FIFO usam filas FIFO e as de tópicos padrão usam filas padrão.

## Pré-requisitos

Antes de configurar uma fila de mensagens mortas, complete os seguintes pré-requisitos:

1. [Crie um tópico do Amazon SNS](#) chamado MyTopic.
2. [Crie uma fila do Amazon SQS](#) chamada MyEndpoint, para ser usada como endpoint para a assinatura do Amazon SNS.
3. (Ignorar para AWS CloudFormation) [Inscreva a fila do tópico](#).
4. [Crie outra fila do Amazon SQS](#) chamada MyDeadLetterQueue, para ser usada como a fila de mensagens mortas para a assinatura do Amazon SNS.
5. Para conceder à entidade principal do Amazon SNS acesso à ação da API do Amazon SQS, defina a seguinte política de fila para MyDeadLetterQueue.

```
{
  "Statement": [{
    "Effect": "Allow",
```

```
"Principal": {
  "Service": "sns.amazonaws.com"
},
"Action": "SQS:SendMessage",
"Resource": "arn:aws:sqs:us-east-2:123456789012:MyDeadLetterQueue",
"Condition": {
  "ArnEquals": {
    "aws:SourceArn": "arn:aws:sns:us-east-2:123456789012:MyTopic"
  }
}
}]
}
```

## Tópicos

- [Para configurar uma fila de mensagens mortas para uma assinatura do Amazon SNS usando o AWS Management Console](#)
- [Para configurar uma fila de mensagens mortas para uma assinatura do Amazon SNS usando um SDK AWS](#)
- [Para configurar uma fila de mensagens mortas para uma assinatura do Amazon SNS usando o AWS CLI](#)
- [Para configurar uma fila de mensagens mortas para uma assinatura do Amazon SNS usando AWS CloudFormation](#)

Para configurar uma fila de mensagens mortas para uma assinatura do Amazon SNS usando o AWS Management Console

Verifique os [pré-requisitos](#) antes de começar este tutorial.

1. Faça login no [console do Amazon SQS](#).
2. [Crie uma fila do Amazon SQS](#) ou use uma fila existente e anote o ARN da fila na guia Details (Detalhes) da fila, por exemplo:

```
arn:aws:sqs:us-east-2:123456789012:MyDeadLetterQueue
```

3. Faça login no [console do Amazon SNS](#).
4. No painel de navegação, escolha Subscriptions (Assinaturas).
5. Na página Subscriptions (Inscrições) selecione uma inscrição existente e escolha Edit (Editar).

6. Na página Edit **1234a567-bc89-012d-3e45-6fg7h890123i** (Editar 1234a567-bc89-012d-3e45-6fg7h890123i), expanda a seção Redrive policy (dead-letter queue) (Política de redirecionamento (fila de mensagens mortas) e faça o seguinte:
  - a. Selecione Ativado.
  - b. Especifique o ARN de uma fila do Amazon SQS.
7. Escolha Salvar alterações.

A inscrição estará configurada para usar uma fila de mensagens mortas.

## Para configurar uma fila de mensagens mortas para uma assinatura do Amazon SNS usando um SDK AWS

Antes de executar este exemplo, verifique se você preencheu os [pré-requisitos](#).

Para usar um AWS SDK, você deve configurá-lo com suas credenciais. Para obter mais informações, consulte [The shared config and credentials files](#) (“Os arquivos compartilhados de configuração e de credenciais”) no Guia de referência de AWS SDKs e ferramentas.

O exemplo de código a seguir mostra como usar `setSubscriptionAttributesRedrivePolicy`.

### Java

#### SDK para Java 1.x

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```
// Specify the ARN of the Amazon SNS subscription.
String subscriptionArn =
    "arn:aws:sns:us-east-2:123456789012:MyEndpoint:1234a567-
bc89-012d-3e45-6fg7h890123i";

// Specify the ARN of the Amazon SQS queue to use as a dead-letter queue.
String redrivePolicy =
    "{\"deadLetterTargetArn\":\"arn:aws:sqs:us-
east-2:123456789012:MyDeadLetterQueue\"}";
```

```
// Set the specified Amazon SQS queue as a dead-letter queue
// of the specified Amazon SNS subscription by setting the RedrivePolicy
attribute.
SetSubscriptionAttributesRequest request = new SetSubscriptionAttributesRequest()
    .withSubscriptionArn(subscriptionArn)
    .withAttributeName("RedrivePolicy")
    .withAttributeValue(redrivePolicy);
sns.setSubscriptionAttributes(request);
```

Para configurar uma fila de mensagens mortas para uma assinatura do Amazon SNS usando o AWS CLI

Verifique os [pré-requisitos](#) antes de começar este tutorial.

1. Instale e configure a AWS CLI. Para mais informações, consulte o [Guia do usuário do AWS Command Line Interface](#).
2. Use o comando a seguir.

```
aws sns set-subscription-attributes \
--subscription-arn arn:aws:sns:us-east-2:123456789012:MyEndpoint:1234a567-
bc89-012d-3e45-6fg7h890123i
--attribute-name RedrivePolicy
--attribute-value "{\"deadLetterTargetArn\": \"arn:aws:sqs:us-
east-2:123456789012:MyDeadLetterQueue\"}"
```

Para configurar uma fila de mensagens mortas para uma assinatura do Amazon SNS usando AWS CloudFormation

Verifique os [pré-requisitos](#) antes de começar este tutorial.

1. Copie o seguinte código JSON em um arquivo denominado `MyDeadLetterQueue.json`.

```
{
  "Resources": {
    "mySubscription": {
      "Type" : "AWS::SNS::Subscription",
      "Properties" : {
```

```
"Protocol": "sqs",
"Endpoint": "arn:aws:sqs:us-east-2:123456789012:MyEndpoint",
"TopicArn": "arn:aws:sns:us-east-2:123456789012:MyTopic",
"RedrivePolicy": {
  "deadLetterTargetArn":
    "arn:aws:sqs:us-east-2:123456789012:MyDeadLetterQueue"
}
}
}
}
```

2. Faça login no [AWS CloudFormation console](#).
3. Na página Select Template (Selecionar modelo), selecione Upload a template to Amazon S3 (Fazer upload de um modelo no Amazon S3), selecione o arquivo MyDeadLetterQueue.json e escolha Next (Próximo).
4. Na página Specify Details (Especificar detalhes), digite MyDeadLetterQueue em Stack Name (Nome da pilha) e escolha Next (Próximo).
5. Na página Options (Opções), escolha Next (Avançar).
6. Na página Review (Revisar), escolha Create (Criar).

AWS CloudFormation começa a criar a MyDeadLetterQueue pilha e exibe o status CREATE\_IN\_PROGRESS. Quando o processo estiver concluído, AWS CloudFormation exibirá o status CREATE\_COMPLETE.

# Arquivamento, reprodução e análise de mensagens do Amazon SNS

Os tópicos padrão do Amazon SNS oferecem suporte ao arquivamento de mensagens por meio do Amazon Data Firehose. Você pode distribuir notificações para os fluxos de entrega do Firehose, o que permite enviar notificações para destinos de armazenamento e análise compatíveis com o Firehose, incluindo Amazon Simple Storage Service (Amazon S3), Amazon Redshift e muito mais.

Os tópicos FIFO do Amazon SNS comportam um arquivamento de mensagens local, sem código, que permite aos proprietários de tópicos armazenar (ou arquivar) mensagens publicadas em um tópico por até 365 dias. No caso de tópicos com uma `ArchivePolicy` ativa, os assinantes podem então criar uma `ReplayPolicy` para recuperar (ou reproduzir) as mensagens arquivadas em um endpoint inscrito. Para saber mais sobre esse atributo, consulte [Arquivamento e reprodução de mensagens de tópicos FIFO](#).

Atributos	Tópicos padrão	Tópicos FIFO
Arquivamento de mensagens	<a href="#">Fanout para streams de entrega do Firehose</a>	<a href="#">Arquivamento de mensagens para proprietários de tópicos FIFO</a>
Reprodução de mensagens	A reprodução de tópicos padrão não é um recurso incorporado. Muitos clientes criam o próprio recurso com base no arquivo de mensagens.	<a href="#">Reprodução de mensagens para assinantes de tópicos FIFO</a>

# Usar o Amazon SNS para sistemas de mensagens de aplicação para aplicação (A2A)

Esta seção fornece informações sobre o uso do Amazon SNS para sistemas de mensagens de aplicação para aplicação com assinantes.

## Tópicos

- [Fanout para streams de entrega do Firehose](#)
- [Fanout para funções do Lambda](#)
- [Fanout para filas do Amazon SQS](#)
- [Fanout para endpoints HTTP\(S\)](#)
- [Fanout para AWS Event Fork Pipelines](#)
- [Usar o Agendador do Amazon EventBridge com o Amazon SNS](#)

## Fanout para streams de entrega do Firehose

Você pode inscrever [streams de entrega do Amazon Data Firehose para tópicos do Amazon SNS](#), o que permite enviar notificações para endpoints adicionais de armazenamento e análise. As mensagens publicadas em um tópico do Amazon SNS são enviadas para o stream de entrega inscrito do Firehose e entregues ao destino conforme configurado no Firehose. O proprietário de uma assinatura pode inscrever até cinco streams de entrega do Firehose para um tópico do Amazon SNS. Cada stream de entrega do Firehose tem uma [cota padrão](#) para solicitações e taxa de transferência por segundo. Esse limite poderia resultar em mais mensagens publicadas (tráfego de entrada) do que entregues (tráfego de saída). Se houver mais tráfego de entrada que de saída, sua assinatura pode acumular um grande backlog de mensagens, causando alta latência na entrega de mensagens. Você pode solicitar um [aumento na cota](#) com base na taxa de publicação para evitar impactos adversos na workload.

Por meio dos streams de entrega do Firehose, você pode distribuir notificações do Amazon SNS para o Amazon Simple Storage Service (Amazon S3), Amazon Redshift, OpenSearch Amazon Service (Service) e para provedores de serviços terceirizados, como OpenSearch Datadog, New Relic, MongoDB e Splunk.

Por exemplo, você pode usar essa funcionalidade para armazenar permanentemente mensagens enviadas para um tópico em um bucket do Amazon S3 para fins de conformidade, arquivamento

ou outros. Para fazer isso, crie um stream de entrega do Firehose com um destino de bucket S3 e inscreva esse stream de entrega no tópico do Amazon SNS. Como outro exemplo, para realizar análises em mensagens enviadas para um tópico do Amazon SNS, crie um stream de entrega com um destino de índice OpenSearch de serviços. Em seguida, você pode inscrever o stream de entrega do Firehose no tópico Amazon SNS.

O Amazon SNS também oferece suporte ao registro do status de entrega de mensagens para notificações enviadas aos endpoints do Firehose. Para ter mais informações, consulte [Status de entrega de mensagens do Amazon SNS](#).

## Tópicos

- [Pré-requisitos para assinar streams de entrega do Firehose para tópicos do Amazon SNS](#)
- [Inscrever um stream de entrega do Firehose em um tópico do Amazon SNS](#)
- [Trabalhar com destinos de fluxo de entrega](#)
- [Exemplo de caso de uso para arquivamento e análise de mensagens](#)

## Pré-requisitos para assinar streams de entrega do Firehose para tópicos do Amazon SNS

Para assinar um stream de entrega do Amazon Data Firehose para um tópico do SNS, você Conta da AWS deve ter:

- Um tópico do SNS padrão. Para ter mais informações, consulte [Criar um tópico do Amazon SNS](#).
- Um stream de entrega do Firehose. Para obter mais informações, consulte [Como criar um stream de entrega do Amazon Data Firehose](#) e [conceder ao seu aplicativo acesso aos seus recursos do Firehose no Guia do desenvolvedor](#) do Amazon Data Firehose.
- Uma função do AWS Identity and Access Management(IAM) que confia no principal de serviço do Amazon SNS e tem permissão para gravar no fluxo de entrega. Você deverá inserir o nome de recurso da Amazon (ARN) dessa função como `SubscriptionRoleARN` quando cria a assinatura. O Amazon SNS assume essa função, o que permite que o Amazon SNS coloque registros no stream de entrega do Firehose.

O exemplo de política seguinte mostra as permissões recomendadas:

```
{
  "Version": "2012-10-17",
  "Statement": [
```

```

{
  "Action": [
    "firehose:DescribeDeliveryStream",
    "firehose:ListDeliveryStreams",
    "firehose:ListTagsForDeliveryStream",
    "firehose:PutRecord",
    "firehose:PutRecordBatch"
  ],
  "Resource": [
    "arn:aws:firehose:us-east-1:111111111111:deliverystream/firehose-sns-delivery-stream"
  ],
  "Effect": "Allow"
}
]
}

```

Para fornecer permissão total para usar o Firehose, você também pode usar a política AWS gerenciada. `AmazonKinesisFirehoseFullAccess` Ou, para fornecer permissões mais rígidas para o uso do Firehose, você pode criar sua própria política. No mínimo, a política deve fornecer permissão para executar a operação `PutRecord` em um fluxo de entrega específico.

Em todos os casos, você também deve editar o relacionamento de confiança para incluir o principal de serviço do Amazon SNS. Por exemplo: .

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "sns.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}

```

Para obter mais informações sobre como criar uma função, consulte [Criar uma função para delegar permissões a um serviço da AWS](#) no Manual do usuário do IAM.

Depois de concluir esses requisitos, você pode [assinar o fluxo de entrega do tópico do SNS](#).

## Inscrever um stream de entrega do Firehose em um tópico do Amazon SNS

[Para entregar notificações do Amazon SNS aos fluxos de entrega do Amazon Data Firehose, primeiro certifique-se de que você atendeu a todos os pré-requisitos.](#) Para obter uma lista dos endpoints compatíveis, consulte os [endpoints e cotas do Amazon Data Firehose](#) no. Referência geral da Amazon Web Services

Para inscrever um stream de distribuição do Firehose em um tópico

1. Faça login no [console do Amazon SNS](#).
2. No painel de navegação, escolha Subscriptions.
3. Na página Subscriptions (Assinaturas), escolha Create subscription (Criar assinatura).
4. Na página Create subscription (Criar inscrição), na seção Details (detalhes), faça o seguinte:
  - a. Em Topic ARN (ARN do tópico), escolha o nome do recurso da Amazon (ARN) de um tópico padrão.
  - b. Em Protocolo, escolha Firehose.
  - c. Para Endpoint, escolha o ARN de um stream de entrega do Firehose que pode receber notificações do Amazon SNS.
  - d. Para ARN da função de assinatura, especifique o ARN da função AWS Identity and Access Management (IAM) que você criou para gravar nos fluxos de entrega do Firehose. Para ter mais informações, consulte [Pré-requisitos para assinar streams de entrega do Firehose para tópicos do Amazon SNS](#).
  - e. (Opcional) Para remover quaisquer metadados do Amazon SNS das mensagens publicadas, escolha Enable raw message delivery (Habilitar a entrega de mensagens). Para ter mais informações, consulte [Entrega de mensagens brutas do Amazon SNS](#).
5. (Opcional) Para configurar uma política de filtros, expanda a seção Subscription filter policy (Política de filtro de assinatura). Para ter mais informações, consulte [Políticas de filtro de assinatura do Amazon SNS](#).
6. (Opcional) Para configurar uma fila de mensagens mortas para a assinatura, expanda a seção Redrive policy (dead-letter queue) (Política de redirecionamento (fila de mensagens mortas)). Para ter mais informações, consulte [Filas de mensagens mortas \(DLQs\) do Amazon SNS](#).
7. Selecione Criar assinatura.

O console cria a assinatura e abre a página Details (Detalhes) da assinatura.

## Trabalhar com destinos de fluxo de entrega

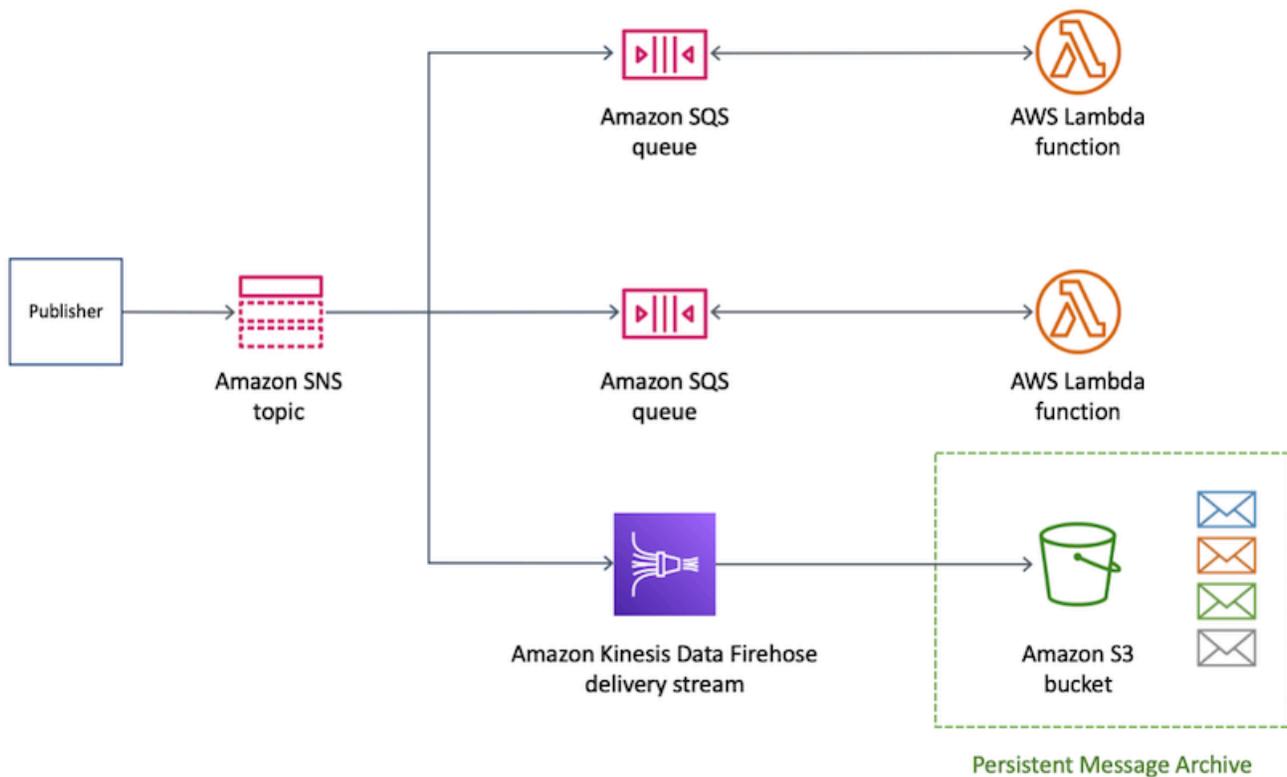
Por meio dos [streams de entrega do Amazon Data Firehose](#), você pode enviar mensagens para endpoints adicionais. Esta seção descreve como trabalhar com destinos com suporte.

### Tópicos

- [Destinos do Amazon S3](#)
- [OpenSearch Destinos de serviços](#)
- [Destinos do Amazon Redshift](#)
- [Destinos HTTP](#)

### Destinos do Amazon S3

Esta seção fornece informações sobre os fluxos de entrega do Amazon Data Firehose que publicam dados no Amazon Simple Storage Service (Amazon S3).



### Tópicos

- [Formato de mensagem arquivada para destinos do Amazon S3](#)
- [Analisar mensagens para destinos do Amazon S3](#)

## Formato de mensagem arquivada para destinos do Amazon S3

O exemplo a seguir mostra uma notificação do Amazon SNS enviada para um bucket do Amazon Simple Storage Service (Amazon S3), usando recuos para legibilidade.

### Note

Neste exemplo, a entrega de mensagens brutas está desativada para a mensagem publicada. Quando a entrega de mensagens brutas é desativada, o Amazon SNS adiciona metadados JSON à mensagem, incluindo estas propriedades:

- Type
- MessageId
- TopicArn
- Subject
- Timestamp
- UnsubscribeURL
- MessageAttributes

Para obter mais informações sobre a entrega de mensagens brutas, consulte [Entrega de mensagens brutas do Amazon SNS](#).

```
{
  "Type": "Notification",
  "MessageId": "719a6bbf-f51b-5320-920f-3385b5e9aa56",
  "TopicArn": "arn:aws:sns:us-east-1:333333333333:my-kinesis-test-topic",
  "Subject": "My 1st subject",
  "Message": "My 1st body",
  "Timestamp": "2020-11-26T23:48:02.032Z",
  "UnsubscribeURL": "https://sns.us-east-1.amazonaws.com/?
Action=Unsubscribe&SubscriptionArn=arn:aws:sns:us-east-1:333333333333:my-kinesis-test-
topic:0b410f3c-ee5e-49d8-b59b-3b4aa6d8fcf5",
  "MessageAttributes": {
```

```

    "myKey1": {
      "Type": "String",
      "Value": "myValue1"
    },
    "myKey2": {
      "Type": "String",
      "Value": "myValue2"
    }
  }
}

```

O exemplo a seguir mostra três mensagens do SNS enviadas por meio de um stream de entrega do Amazon Data Firehose para o mesmo bucket do Amazon S3. O buffer é levado em conta e as quebras de linha separam as mensagens.

```

{"Type":"Notification","MessageId":"d7d2513e-6126-5d77-
bbe2-09042bd0a03a","TopicArn":"arn:aws:sns:us-east-1:333333333333:my-
kinesis-test-topic","Subject":"My 1st subject","Message":"My 1st
body","Timestamp":"2020-11-27T00:30:46.100Z","UnsubscribeURL":"https://
sns.us-east-1.amazonaws.com/?Action=Unsubscribe&SubscriptionArn=arn:aws:sns:us-
east-1:313276652360:my-kinesis-test-topic:0b410f3c-ee5e-49d8-
b59b-3b4aa6d8f5cf5","MessageAttributes":{"myKey1":
{"Type":"String","Value":"myValue1"},"myKey2":{"Type":"String","Value":"myValue2"}}}
{"Type":"Notification","MessageId":"0c0696ab-7733-5bfb-b6db-
ce913c294d56","TopicArn":"arn:aws:sns:us-east-1:333333333333:my-
kinesis-test-topic","Subject":"My 2nd subject","Message":"My 2nd
body","Timestamp":"2020-11-27T00:31:22.151Z","UnsubscribeURL":"https://
sns.us-east-1.amazonaws.com/?Action=Unsubscribe&SubscriptionArn=arn:aws:sns:us-
east-1:313276652360:my-kinesis-test-topic:0b410f3c-ee5e-49d8-
b59b-3b4aa6d8f5cf5","MessageAttributes":{"myKey1":{"Type":"String","Value":"myValue1"}}}
{"Type":"Notification","MessageId":"816cd54d-8cfa-58ad-91c9-8d77c7d173aa","TopicArn":"arn:aws:s
east-1:333333333333:my-kinesis-test-topic","Subject":"My 3rd subject","Message":"My
3rd body","Timestamp":"2020-11-27T00:31:39.755Z","UnsubscribeURL":"https://
sns.us-east-1.amazonaws.com/?Action=Unsubscribe&SubscriptionArn=arn:aws:sns:us-
east-1:313276652360:my-kinesis-test-topic:0b410f3c-ee5e-49d8-b59b-3b4aa6d8f5cf5"}

```

## Analisar mensagens para destinos do Amazon S3

Esta página descreve como analisar mensagens do Amazon SNS enviadas por meio de fluxos de entrega do Amazon Data Firehose para destinos do Amazon Simple Storage Service (Amazon S3).

Para analisar mensagens do SNS enviadas por meio de fluxos de entrega do Firehose para destinos do Amazon S3

1. Configure seus recursos do Amazon S3. Para obter instruções, consulte [Criação de buckets](#) no Manual do usuário do Amazon Simple Storage Service e [Como trabalhar com buckets do Amazon S3](#) no Manual do usuário Amazon Simple Storage Service.
2. Configure seu fluxo de entrega. Para obter instruções, consulte [Escolha o Amazon S3 para seu destino no Guia](#) do desenvolvedor do Amazon Data Firehose.
3. Use o [Amazon Athena](#) para consultar os objetos do Amazon S3 com o SQL padrão. Para obter mais informações, consulte [Conceitos básicos](#) no Manual do usuário do Amazon Athena.

### Consulta de exemplo

Para esta consulta de exemplo, suponha o seguinte:

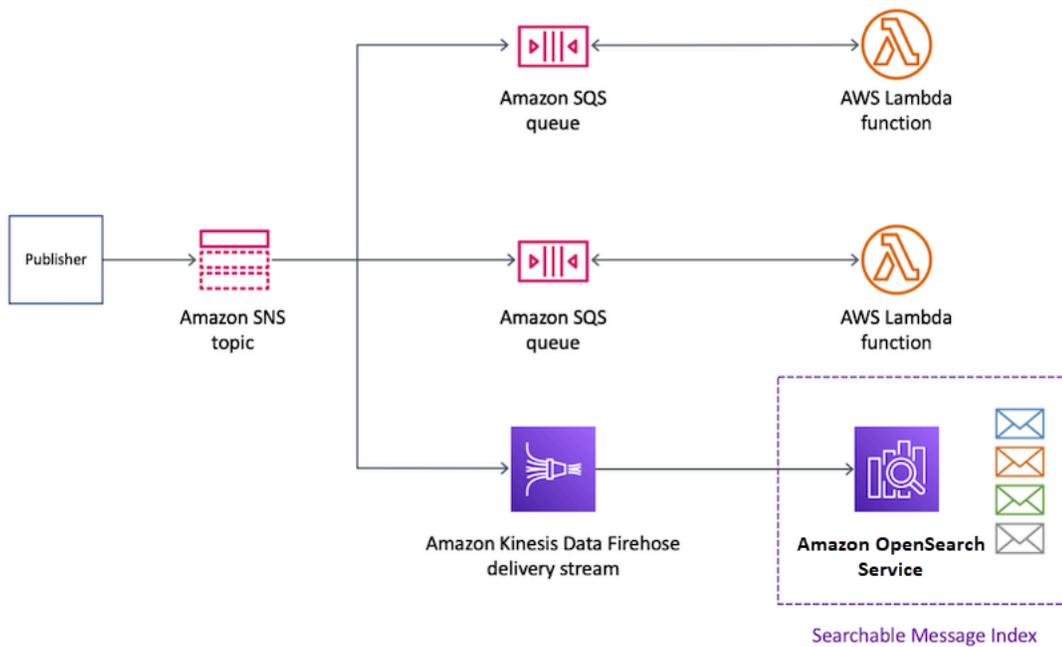
- As mensagens são armazenadas na tabela notifications no esquema default.
- A tabela notifications inclui uma coluna timestamp com um tipo de string.

A consulta a seguir retorna todas as mensagens do SNS recebidas no intervalo de datas especificado:

```
SELECT *
FROM default.notifications
WHERE from_iso8601_timestamp(timestamp) BETWEEN TIMESTAMP '2020-12-01 00:00:00' AND
TIMESTAMP '2020-12-02 00:00:00';
```

### OpenSearch Destinos de serviços

Esta seção fornece informações sobre os fluxos de entrega do Amazon Data Firehose que publicam dados no Amazon OpenSearch Service (Service)OpenSearch .



## Tópicos

- [Formato de mensagens arquivadas em índices do OpenSearch Service](#)
- [Analisando mensagens para destinos OpenSearch de serviços](#)

## Formato de mensagens arquivadas em índices do OpenSearch Service

O exemplo a seguir mostra uma notificação do Amazon SNS enviada para um índice do Amazon OpenSearch Service (OpenSearch Service) chamado `my-index`. Esse índice tem um campo de filtro de tempo no campo `Timestamp`. A notificação SNS é colocada na propriedade `_source` da carga útil.

### Note

Neste exemplo, a entrega de mensagens brutas está desativada para a mensagem publicada. Quando a entrega de mensagens brutas é desativada, o Amazon SNS adiciona metadados JSON à mensagem, incluindo estas propriedades:

- `Type`
- `MessageId`
- `TopicArn`
- `Subject`

- Timestamp
- UnsubscribeURL
- MessageAttributes

Para obter mais informações sobre a entrega de mensagens brutas, consulte [Entrega de mensagens brutas do Amazon SNS](#).

```
{
  "_index": "my-index",
  "_type": "_doc",
  "_id": "49613100963111323203250405402193283794773886550985932802.0",
  "_version": 1,
  "_score": null,
  "_source": {
    "Type": "Notification",
    "MessageId": "bf32e294-46e3-5dd5-a6b3-bad65162e136",
    "TopicArn": "arn:aws:sns:us-east-1:111111111111:my-topic",
    "Subject": "Sample subject",
    "Message": "Sample message",
    "Timestamp": "2020-12-02T22:29:21.189Z",
    "UnsubscribeURL": "https://sns.us-east-1.amazonaws.com/?
Action=Unsubscribe&SubscriptionArn=arn:aws:sns:us-east-1:111111111111:my-
topic:b5aa9bc1-9c3d-452b-b402-aca2cefc63c9",
    "MessageAttributes": {
      "my_attribute": {
        "Type": "String",
        "Value": "my_value"
      }
    }
  },
  "fields": {
    "Timestamp": [
      "2020-12-02T22:29:21.189Z"
    ]
  },
  "sort": [
    1606948161189
  ]
}
```

## Analisando mensagens para destinos OpenSearch de serviços

Esta página descreve como analisar mensagens do Amazon SNS enviadas por meio de fluxos de entrega do Amazon Data Firehose para destinos do Amazon OpenSearch Service (Service). OpenSearch

Para analisar as mensagens do SNS enviadas por meio de fluxos OpenSearch de entrega do Firehose para destinos de serviço

1. Configure seus recursos OpenSearch de serviço. Para obter instruções, consulte [Conceitos básicos do Amazon OpenSearch Service](#) no Amazon OpenSearch Service Developer Guide.
2. Configure seu fluxo de entrega. Para obter instruções, consulte [Escolha o OpenSearch serviço para seu destino](#) no Guia do desenvolvedor do Amazon Data Firehose.
3. Execute uma consulta usando consultas OpenSearch de serviço e Kibana. Para obter mais informações, consulte [Etapa 3: Pesquisar documentos em um domínio de OpenSearch serviço e Kibana](#) no Amazon OpenSearch Service Developer Guide.

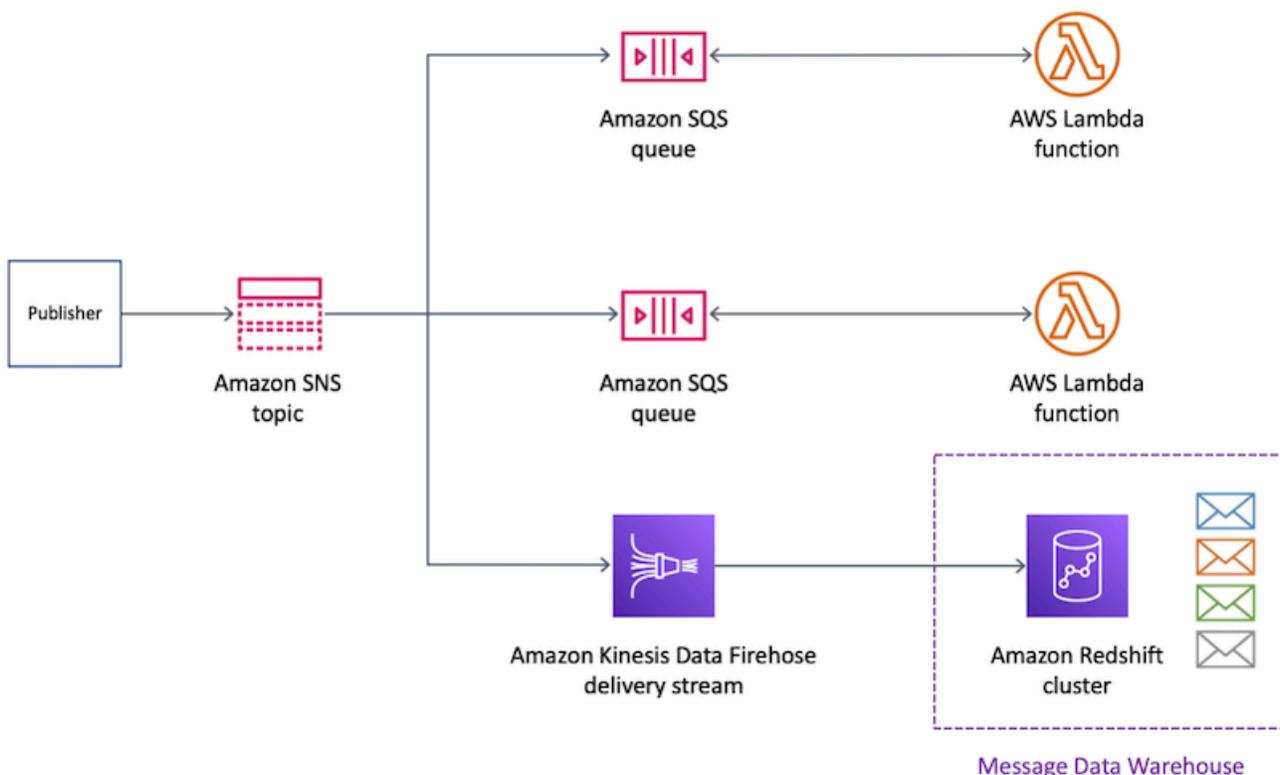
### Consulta de exemplo

O exemplo a seguir cria o índice `my-index` para todas as mensagens do SNS recebidas no intervalo de datas especificado:

```
POST https://search-my-domain.us-east-1.es.amazonaws.com/my-index/_search
{
  "query": {
    "bool": {
      "filter": [
        {
          "range": {
            "Timestamp": {
              "gte": "2020-12-08T00:00:00.000Z",
              "lte": "2020-12-09T00:00:00.000Z",
              "format": "strict_date_optional_time"
            }
          }
        }
      ]
    }
  }
}
```

## Destinos do Amazon Redshift

Esta seção descreve como distribuir notificações do Amazon SNS para um stream de entrega do Amazon Data Firehose que publica dados no Amazon Redshift. Com essa configuração, é possível se conectar ao banco de dados do Amazon Redshift e usar uma ferramenta de consulta SQL para consultar o banco de dados para mensagens do Amazon SNS que atenderem a determinados critérios.



### Tópicos

- [Estrutura da tabela de arquivamento para destinos Amazon Redshift](#)
- [Analisar mensagens para destinos do Amazon Redshift](#)

### Estrutura da tabela de arquivamento para destinos Amazon Redshift

Para endpoints do Amazon Redshift, as mensagens publicadas do Amazon SNS são arquivadas como linhas em uma tabela. Veja um exemplo a seguir.

**Note**

Neste exemplo, a entrega de mensagens brutas está desativada para a mensagem publicada. Quando a entrega de mensagens brutas é desativada, o Amazon SNS adiciona metadados JSON à mensagem, incluindo estas propriedades:

- Type
- MessageId
- TopicArn
- Subject
- Message
- Timestamp
- UnsubscribeURL
- MessageAttributes

Para obter mais informações sobre a entrega de mensagens brutas, consulte [Entrega de mensagens brutas do Amazon SNS](#).

Embora o Amazon SNS adicione propriedades à mensagem usando a capitalização mostrada nesta lista, os nomes de colunas nas tabelas do Amazon Redshift aparecem em todos os caracteres minúsculos. Para transformar os metadados JSON para o endpoint do Amazon Redshift, você pode usar o comando COPY. Para obter mais informações, consulte [Copy from JSON examples](#) (“Copiar de exemplos JSON”) e [Load from JSON data using the 'auto ignorecase' option](#) (“Carregar de dados JSON usando a opção “auto ignorecase””) no Guia do desenvolvedor de banco de dados do Amazon Redshift.

type	messageid	topicarn	subject	mensagem	timestamp	unsubscribeurl	messageattributes
Notificação	ea544832-a0d8-581d-9275-108243c46103	arn:aws:sns:us-east-1:111111111111:my-topic	Exemplo do assunto	Exemplo de mensagem	2020-12-02T00:33:32.272Z	https://sns.us-east-1.amazonaws.com/?	{"my_attribute\":"Type\":"String","\Value\

type	messageid	topicarn	subject	mensagem	timestamp	unsubscribeurl	messageattributes
						Action=Unsubscribe&SubscriptionArn=arn:aws:sns:us-east-1:111111111111:my-topic:326deebc-bf4-45da-b92b-ca77a247813b	e\:"my_value\"}}

type	messageid	topicarn	subject	mensagem	timestamp	unsubscribeurl	messageattributes
Notificação	ab124832-a0d8-581d-9275-108243c46114	arn:aws:sns:us-east-1:111111111111:my-topic	Exemplo de assunto 2	Exemplo de mensagem 2	2020-12-03T00:18:11.129Z	https://sns.us-east-1.amazonaws.com/?Action=Unsubscribe&SubscriptionArn=arn:aws:sns:us-east-1:111111111111:my-topic:326deeeb-cbf4-45da-b92b-ca77a247813b	{"my_attribute2":{"Type":"String","Value":"my_value"}}

type	messageid	topicarn	subject	mensagem	timestamp	unsubscribeurl	messageattributes
Notificação	ce644832-a0d8-581d-9275-108243c46125	arn:aws:sns:us-east-1:111111111111:my-topic	Exemplo de assunto 3	Exemplo de mensagem 3	2020-12-09T00:08:44.405Z	https://sns.us-east-1.amazonaws.com/?Action=Unsubscribe&SubscriptionArn=arn:aws:sns:us-east-1:111111111111:my-topic:326deeeb-cbf4-45da-b92b-ca77a247813b	{"my_attribute3":{"Type":"String","Value":"my_value"}}

Para obter mais informações sobre divulgação de notificações de eventos para o Amazon Redshift, consulte [Destinos do Amazon Redshift](#).

### Analisar mensagens para destinos do Amazon Redshift

Esta página descreve como analisar mensagens do Amazon SNS enviadas por meio de fluxos de entrega do Amazon Data Firehose para destinos do Amazon Redshift.

Para analisar mensagens do SNS enviadas por meio de fluxos de entrega do Firehose para destinos do Amazon Redshift

1. Configure seus recursos do Amazon Redshift. Para obter instruções, consulte [Conceitos básicos do Amazon Redshift](#) no Guia de conceitos básicos do Amazon Redshift.
2. Configure seu fluxo de entrega. Para obter instruções, consulte [Escolha o Amazon Redshift para seu destino no Guia](#) do desenvolvedor do Amazon Data Firehose.
3. Execute uma consulta. Para obter mais informações, consulte [Consultar um banco de dados usando o editor de consultas](#) no Guia de gerenciamento do Amazon Redshift.

### Consulta de exemplo

Para esta consulta de exemplo, suponha o seguinte:

- As mensagens são armazenadas na tabela `notifications` no esquema `public` padrão.
- A propriedade `Timestamp` da mensagem SNS é armazenada na coluna `timestamp` da tabela com um tipo de dados de coluna `timestampz`.

#### Note

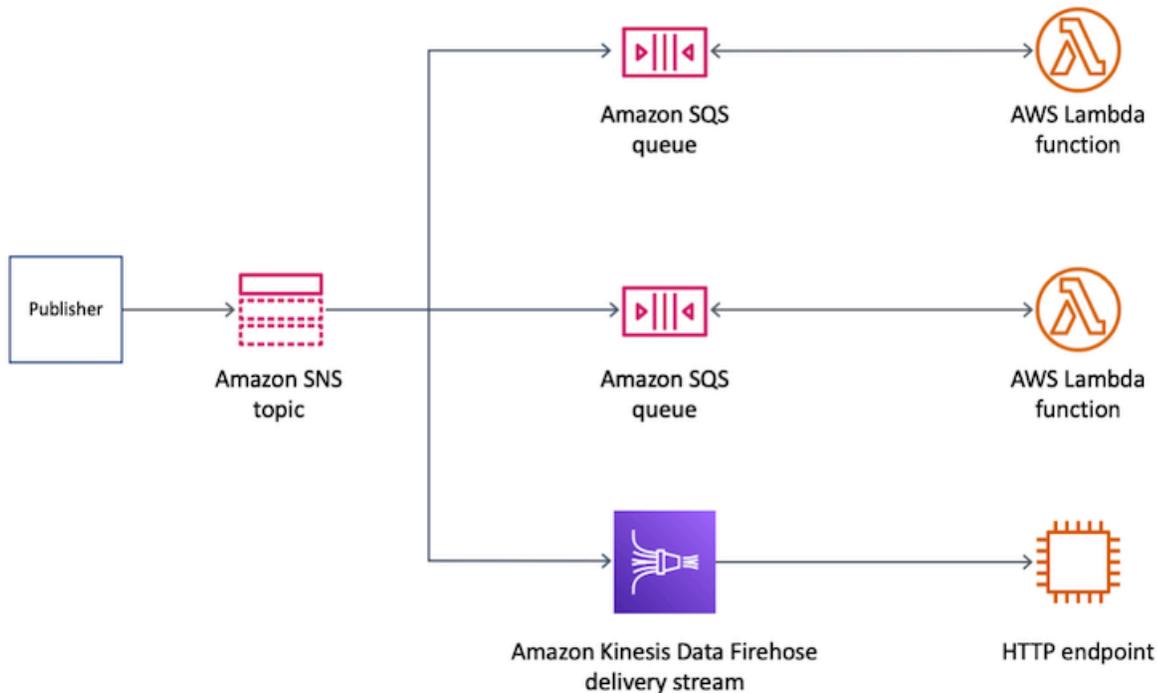
Para transformar os metadados JSON para o endpoint do Amazon Redshift, você pode usar o comando `COPY`. Para obter mais informações, consulte [Copy from JSON examples](#) (“Copiar de exemplos JSON”) e [Load from JSON data using the 'auto ignorecase' option](#) (“Carregar de dados JSON usando a opção “auto ignorecase””) no Guia do desenvolvedor de banco de dados do Amazon Redshift.

A consulta a seguir retorna todas as mensagens do SNS recebidas no intervalo de datas especificado:

```
SELECT *
FROM public.notifications
WHERE timestamp > '2020-12-01T09:00:00.000Z' AND timestamp <
'2020-12-02T09:00:00.000Z';
```

## Destinos HTTP

Esta seção fornece informações sobre os fluxos de entrega do Amazon Data Firehose que publicam dados em endpoints HTTP.



### Tópicos

- [Formato de mensagem entregue para destinos HTTP](#)

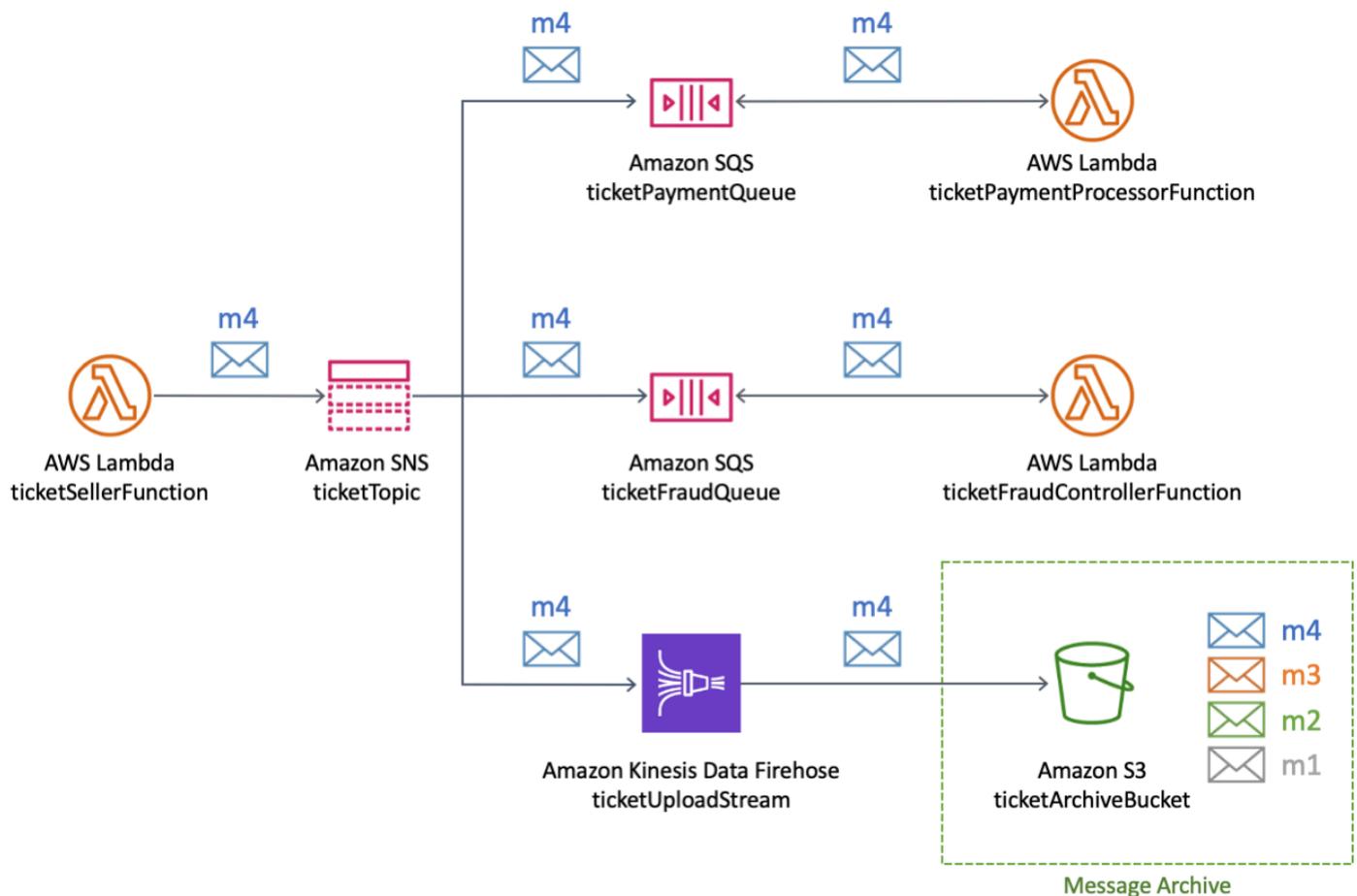
### Formato de mensagem entregue para destinos HTTP

Veja a seguir um exemplo de corpo de solicitação HTTP POST do Amazon SNS que um stream de entrega do Amazon Data Firehose pode enviar para o endpoint HTTP. A notificação SNS é codificada como uma carga base64 na propriedade `records`.

#### **Note**

Neste exemplo, a entrega de mensagens brutas está desativada para a mensagem publicada. Para obter mais informações sobre a entrega de mensagens brutas, consulte [Entrega de mensagens brutas do Amazon SNS](#).





Para executar análises e obter insights sobre vendas de ingressos, a empresa executa consultas SQL usando o Amazon Athena. Por exemplo, a empresa pode consultar para saber mais sobre os destinos mais populares e os passageiros mais frequentes.

Para criar os recursos da AWS para esse caso de uso, você pode usar o AWS Management Console ou um modelo do AWS CloudFormation.

## Tópicos

- [Criar os recursos iniciais](#)
- [Criação do stream de entrega do Firehose](#)
- [Inscrevendo o stream de entrega do Firehose no tópico Amazon SNS](#)
- [Testando e consultando a configuração](#)
- [Usar um modelo do AWS CloudFormation](#)

## Criar os recursos iniciais

Esta página descreve como criar os seguintes recursos para o [caso de uso de exemplo de arquivamento e análise de mensagens](#):

- Um bucket do Amazon Simple Storage Service (Amazon S3)
- Duas filas do Amazon Simple Queue Service (Amazon SQS)
- Um tópico do Amazon SNS
- Duas inscrições do Amazon SQS no tópico do Amazon SNS

Para criar os recursos iniciais

### 1. Criar o bucket do Amazon S3:

- Abra o [console do Amazon S3](#).
- Selecione Create bucket (Criar bucket).
- Em Bucket name (Nome do bucket), insira um nome globalmente exclusivo. Mantenha os outros campos como os padrões.
- Selecione Create bucket (Criar bucket).

Para obter mais informações sobre buckets do Amazon S3, consulte [Criação de buckets](#) no Manual do usuário do Amazon Simple Storage Service e [Como trabalhar com buckets do Amazon S3](#) no Manual do usuário do Amazon Simple Storage Service.

### 2. Crie as duas filas do Amazon SQS:

- Abra o [console do Amazon SQS](#).
- Escolha Create queue.
- Em Type (Tipo), escolha Standard (Padrão).
- Em Name (Nome), insira **ticketPaymentQueue**.
- Em Access policy (Política de acesso), para Choose method (Escolher método), escolha Advanced (Avançado).
- Na caixa JSON policy (Política de JSON), cole a seguinte política:

```
{
  "Version": "2008-10-17",
  "Statement": [
```

```
{
  "Effect": "Allow",
  "Principal": {
    "Service": "sns.amazonaws.com"
  },
  "Action": "sqs:SendMessage",
  "Resource": "*",
  "Condition": {
    "ArnEquals": {
      "aws:SourceArn": "arn:aws:sns:us-east-1:123456789012:ticketTopic"
    }
  }
}
```

Nesta política de acesso, substitua o número da Conta da AWS (*123456789012*) pelo seu próprio e altere a região da AWS (*us-east-1*) da maneira adequada.

- g. Escolha Create queue.
- h. Repita essas etapas para criar uma segunda fila do SQS chamada **ticketFraudQueue**.

Para obter mais informações sobre a criação de filas do SQS, consulte [Creating an Amazon SQS queue \(console\)](#) no Guia do desenvolvedor do Amazon Simple Queue Service.

3. Criar o tópico do SNS:
  - a. Abra a [página Topics](#) (Tópicos) no console do Amazon SNS.
  - b. Escolha Create topic.
  - c. Em Details (Detalhes), para Type (Tipo), escolha Standard (Padrão).
  - d. Em Name (Nome), insira **ticketTopic**.
  - e. Escolha Create topic.

Para obter mais informações sobre a criação de tópicos do SNS, consulte [Criar um tópico do Amazon SNS](#).

4. Inscreva as duas filas do SQS no tópico do SNS:
  - a. No [console do Amazon SNS](#), na página de detalhes do tópico ticketTopic, escolha Create subscription (Criar assinatura).

- b. Em Details (Detalhes), para Protocol (Protocolo), escolha Amazon SQS.
- c. Para Endpoint escolha o nome do recurso da Amazon (ARN) da fila ticketPaymentQueue.
- d. Selecione Create subscription.
- e. Repita essas etapas para criar uma segunda assinatura usando o ARN da fila ticketFraudQueue.

Para obter mais informações sobre a inscrição em tópicos do SNS, consulte [Assinar tópico do Amazon SNS](#). Você também pode assinar filas SQS para tópicos do SNS no console do Amazon SQS. Para obter mais informações, consulte [Subscribing an Amazon SQS queue to an Amazon SNS topic \(console\)](#) no Guia do desenvolvedor do Amazon Simple Queue Service.

Você criou os recursos iniciais para este exemplo de caso de uso. Para continuar, consulte [Criação do stream de entrega do Firehose](#).

## Criação do stream de entrega do Firehose

Esta página descreve como criar o stream de entrega do Amazon Data Firehose para o [exemplo de caso de uso de arquivamento e análise de mensagens](#).

Para criar o stream de entrega do Firehose

1. Abra o [console de serviços do Amazon Kinesis](#).
2. Escolha Firehose e, em seguida, escolha Criar fluxo de entrega.
3. Na página New delivery stream (Novo fluxo de entrega), em Delivery stream name, (Nome do fluxo de entrega) insira **ticketUploadStream** e, depois, escolha Next (Próximo).
4. Na página Process records (Processar registros), escolha Next Step (Próxima etapa).
5. Na página Choose a Destination (Selecionar um destino), faça o seguinte:
  - a. Para Destination (Destino), escolha Amazon S3.
  - b. Em S3 destination (Destino do S3), em S3 bucket (Bucket do S3), escolha o bucket do S3 que você [criou inicialmente](#).
  - c. Escolha Próximo.
6. Na página Configure settings (Definir configurações), para S3 buffer conditions (Condições de buffer do S3), faça o seguinte:

- Em Buffer size (Tamanho do buffer), insira **1**.
- Em Buffer interval (Intervalo buffer), insira **60**.

O uso desses valores para o buffer do Amazon S3 permite testar rapidamente a configuração. A primeira condição que é atendida aciona a entrega de dados para o bucket do S3.

7. Na página Configure settings (Definir configurações), em Permissions (Permissões), opte por criar uma função do AWS Identity and Access Management (IAM) com as permissões necessárias atribuídas automaticamente. Em seguida, escolha Próximo.
8. Na página Review (Revisar), selecione Create delivery stream (Criar fluxo de entrega).
9. Na página de streams de entrega do Kinesis Data Firehose, escolha o stream de entrega que você acabou de criar (). ticketUploadStream Na guia Details (Detalhes), anote o nome do recurso da Amazon (ARN) do fluxo para mais tarde.

Para obter mais informações sobre a criação de fluxos de entrega, consulte [Criação de um stream de entrega do Amazon Data Firehose](#) no Guia do desenvolvedor do Amazon Data Firehose. Para obter mais informações sobre como criar funções do IAM, consulte [Criar uma função para delegar permissões a um serviço da AWS](#) no Manual do usuário do IAM.

Você criou o stream de entrega do Firehose com as permissões necessárias. Para continuar, consulte [Inscrevendo o stream de entrega do Firehose no tópico Amazon SNS](#).

## Inscrevendo o stream de entrega do Firehose no tópico Amazon SNS

Esta página descreve como criar os seguintes para o [caso de uso de exemplo de arquivamento e análise de mensagens](#):

- A função AWS Identity and Access Management (IAM) que permite que a assinatura do Amazon SNS coloque registros no stream de entrega do Amazon Data Firehose
- A assinatura do stream de entrega do Firehose para o tópico SNS

Para criar a função do IAM para a assinatura do Amazon SNS

1. Abra a página [Roles](#) (Funções) no console do IAM.
2. Selecione Create role.
3. Em Selecionar tipo de entidade confiável, selecione serviço AWS.

4. Para Choose a use case (Escolher um caso de uso), escolha SNS. Então, escolha Próximo: Permissões.
5. Escolha Próximo: etiquetas.
6. Escolha Próximo: revisar.
7. Na página Review (Revisar), em Role name (Nome da função), insira **ticketUploadStreamSubscriptionRole**. Então, escolha Criar perfil.
8. Quando a função for criada, escolha seu nome (ticketUploadStreamSubscriptionRole).
9. Na página Summary (Resumo), escolha Add inline policy (Adicionar política inline).
10. Na página Create policy (Criar política), escolha a guia JSON e cole a seguinte política JSON na caixa:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "firehose:DescribeDeliveryStream",
        "firehose:ListDeliveryStreams",
        "firehose:ListTagsForDeliveryStream",
        "firehose:PutRecord",
        "firehose:PutRecordBatch"
      ],
      "Resource": [
        "arn:aws:firehose:us-east-1:123456789012:deliverystream/
ticketUploadStream"
      ],
      "Effect": "Allow"
    }
  ]
}
```

Nesta política, substitua o número da Conta da AWS (**123456789012**) pelo seu próprio e altere a região da AWS (**us-east-1**) da maneira adequada.

11. Escolha Revisar política.
12. Na página Review policy (Revisar política), em Name (Nome), insira **FirehoseSnsPolicy**. Selecione Criar política.
13. Na página Summary (Resumo) da função, anote o Role ARN (ARN da função) para mais tarde.

Para obter mais informações sobre como criar funções do IAM, consulte [Criar uma função para delegar permissões a um serviço da AWS](#) no Manual do usuário do IAM.

Para inscrever o stream de entrega do Firehose no tópico SNS

1. Abra a [página Topics](#) (Tópicos) no console do Amazon SNS.
2. Na guia Subscriptions (Assinaturas), escolha Create subscription (Criar assinatura).
3. Em Detalhes, em Protocolo, escolha Amazon Data Firehose.
4. Para Endpoint, insira o Amazon Resource Name (ARN) ticketUploadStreamdo stream de entrega que você criou anteriormente. Por exemplo, digite **arn:aws:firehose:us-east-1:123456789012:deliverystream/ticketUploadStream**.
5. Em ARN da função de assinatura, insira o ARN da função do ticketUploadStreamSubscriptionRoleIAM que você criou anteriormente. Por exemplo, digite **arn:aws:iam::123456789012:role/ticketUploadStreamSubscriptionRole**.
6. Marque a caixa de seleção Enable raw message delivery (Habilitar a entrega de mensagens).
7. Selecione Criar assinatura.

Você criou a função do IAM e a assinatura do tópico do SNS. Para continuar, consulte [Testando e consultando a configuração](#).

## Testando e consultando a configuração

Esta página descreve como testar o [caso de uso de exemplo de arquivamento e análise de mensagens](#) publicando uma mensagem no tópico do Amazon SNS. As instruções incluem uma consulta de exemplo que você pode executar e se adaptar às suas próprias necessidades.

Para testar sua configuração

1. Abra a [página Topics](#) (Tópicos) no console do Amazon SNS.
2. Selecione o tópico **ticketTopic**.
3. Selecione Publish message (Publicar mensagem).
4. Na página Publicar mensagem no tópico, insira o seguinte para o corpo da mensagem. Adicione um caractere de nova linha no final da mensagem.

```
{"BookingDate":"2020-12-15","BookingTime":"2020-12-15
04:15:05","Destination":"Miami","FlyingFrom":"Vancouver","TicketNumber":"abcd1234"}
```

Mantenha todas as outras opções como seus valores padrão.

5. Selecione Publish message (Publicar mensagem).

Para obter mais informações sobre publicação de mensagens, consulte [Publicação de mensagens do Amazon SNS](#).

6. Após o intervalo de fluxo de entrega de 60 segundos, abra a caixa de diálogo [Console do Amazon Simple Storage Service \(Amazon S3\)](#) e escolha o bucket do Amazon S3 que você [criou inicialmente](#).

A mensagem publicada é exibida no bucket.

Para consultar os dados

1. Abra o [console do Amazon Athena](#).
2. Execute uma consulta.

Por exemplo, suponha que a tabela notifications no esquema default contenha os seguintes dados:

```
{"BookingDate":"2020-12-15","BookingTime":"2020-12-15
04:15:05","Destination":"Miami","FlyingFrom":"Vancouver","TicketNumber":"abcd1234"}
{"BookingDate":"2020-12-15","BookingTime":"2020-12-15
11:30:15","Destination":"Miami","FlyingFrom":"Omaha","TicketNumber":"efgh5678"}
{"BookingDate":"2020-12-15","BookingTime":"2020-12-15
3:30:10","Destination":"Miami","FlyingFrom":"NewYork","TicketNumber":"ijkl9012"}
{"BookingDate":"2020-12-15","BookingTime":"2020-12-15
12:30:05","Destination":"Delhi","FlyingFrom":"Omaha","TicketNumber":"mnop3456"}
```

Para localizar o destino superior, execute a seguinte consulta:

```
SELECT destination
FROM default.notifications
GROUP BY destination
ORDER BY count(*) desc
LIMIT 1;
```

Para consultar tickets vendidos durante um intervalo de data e hora específico, execute uma consulta como a seguinte:

```
SELECT *
FROM default.notifications
WHERE bookingtime
  BETWEEN TIMESTAMP '2020-12-15 10:00:00'
  AND TIMESTAMP '2020-12-15 12:00:00';
```

Você pode adaptar ambas as consultas de exemplo para suas próprias necessidades. Para obter mais informações sobre como usar o Athena para executar consultas, consulte [Conceitos básicos](#) no Manual do usuário do Amazon Athena.

## Liberar

Para evitar incorrer em cobranças de uso depois de terminar o teste, exclua os seguintes recursos criados durante o tutorial:

- Assinaturas do Amazon SNS
- Tópico do Amazon SNS
- Filas do Amazon Simple Queue Service (Amazon SQS)
- Bucket do Amazon S3
- Stream de entrega do Amazon Data Firehose
- Funções e políticas do AWS Identity and Access Management (IAM)

## Usar um modelo do AWS CloudFormation

Para automatizar a implantação do [caso de uso de exemplo de arquivamento e análise de mensagens](#) do Amazon SNS, você pode usar o seguinte modelo YAML:

```
---
AWSTemplateFormatVersion: '2010-09-09'
Description: Template for creating an SNS archiving use case
Resources:
  ticketUploadStream:
    DependsOn:
      - ticketUploadStreamRolePolicy
    Type: AWS::KinesisFirehose::DeliveryStream
    Properties:
      S3DestinationConfiguration:
        BucketARN: !Sub 'arn:${AWS::Partition}:s3:::${ticketArchiveBucket}'
```

```
    BufferingHints:
      IntervalInSeconds: 60
      SizeInMBs: 1
      CompressionFormat: UNCOMPRESSED
      RoleARN: !GetAtt ticketUploadStreamRole.Arn
ticketArchiveBucket:
  Type: AWS::S3::Bucket
ticketTopic:
  Type: AWS::SNS::Topic
ticketPaymentQueue:
  Type: AWS::SQS::Queue
ticketFraudQueue:
  Type: AWS::SQS::Queue
ticketQueuePolicy:
  Type: AWS::SQS::QueuePolicy
  Properties:
    PolicyDocument:
      Statement:
        Effect: Allow
        Principal:
          Service: sns.amazonaws.com
        Action:
          - sqs:SendMessage
        Resource: '*'
        Condition:
          ArnEquals:
            aws:SourceArn: !Ref ticketTopic
  Queues:
    - !Ref ticketPaymentQueue
    - !Ref ticketFraudQueue
ticketUploadStreamSubscription:
  Type: AWS::SNS::Subscription
  Properties:
    TopicArn: !Ref ticketTopic
    Endpoint: !GetAtt ticketUploadStream.Arn
    Protocol: firehose
    SubscriptionRoleArn: !GetAtt ticketUploadStreamSubscriptionRole.Arn
ticketPaymentQueueSubscription:
  Type: AWS::SNS::Subscription
  Properties:
    TopicArn: !Ref ticketTopic
    Endpoint: !GetAtt ticketPaymentQueue.Arn
    Protocol: sqs
ticketFraudQueueSubscription:
```

```
Type: AWS::SNS::Subscription
Properties:
  TopicArn: !Ref ticketTopic
  Endpoint: !GetAtt ticketFraudQueue.Arn
  Protocol: sqs
ticketUploadStreamRole:
  Type: AWS::IAM::Role
  Properties:
    AssumeRolePolicyDocument:
      Version: '2012-10-17'
      Statement:
        - Sid: ''
          Effect: Allow
          Principal:
            Service: firehose.amazonaws.com
          Action: sts:AssumeRole
ticketUploadStreamRolePolicy:
  Type: AWS::IAM::Policy
  Properties:
    PolicyName: FirehoseTicketUploadStreamRolePolicy
    PolicyDocument:
      Version: '2012-10-17'
      Statement:
        - Effect: Allow
          Action:
            - s3:AbortMultipartUpload
            - s3:GetBucketLocation
            - s3:GetObject
            - s3:ListBucket
            - s3:ListBucketMultipartUploads
            - s3:PutObject
          Resource:
            - !Sub 'arn:aws:s3:::${ticketArchiveBucket}'
            - !Sub 'arn:aws:s3:::${ticketArchiveBucket}/*'
    Roles:
      - !Ref ticketUploadStreamRole
ticketUploadStreamSubscriptionRole:
  Type: AWS::IAM::Role
  Properties:
    AssumeRolePolicyDocument:
      Version: '2012-10-17'
      Statement:
        - Effect: Allow
          Principal:
```

```
Service:
  - sns.amazonaws.com
Action:
  - sts:AssumeRole
Policies:
- PolicyName: SNSKinesisFirehoseAccessPolicy
  PolicyDocument:
    Version: '2012-10-17'
    Statement:
      - Action:
          - firehose:DescribeDeliveryStream
          - firehose:ListDeliveryStreams
          - firehose:ListTagsForDeliveryStream
          - firehose:PutRecord
          - firehose:PutRecordBatch
        Effect: Allow
        Resource:
          - !GetAtt ticketUploadStream.Arn
```

## Fanout para funções do Lambda

O Amazon SNS e o AWS Lambda são integrados para que você possa chamar funções do Lambda com notificações do Amazon SNS. Quando uma mensagem é publicada em um tópico do SNS que tem uma função do Lambda inscrita, a função do Lambda é chamada com a carga útil da mensagem publicada. A função do Lambda recebe a carga útil da mensagem como um parâmetro de entrada e pode manipular as informações na mensagem, publicar a mensagem em outros tópicos do SNS ou enviar a mensagem a outros serviços da AWS.

O Amazon SNS também oferece suporte a atributos de status de entrega para notificações por mensagens enviadas a endpoints do Lambda. Para obter mais informações, consulte [Status de entrega de mensagens do Amazon SNS](#).

## Prerequisites

Para chamar funções do Lambda usando notificações do Amazon SNS, você precisa do seguinte:

- Função Lambda
- Tópico do Amazon SNS

Para obter mais informações sobre como criar uma função do Lambda para ser usada com o Amazon SNS, consulte [Como usar o AWS Lambda com o Amazon Simple Notification Service](#). Para obter informações sobre a criação de um tópico do Amazon SNS, consulte [Criar um tópico](#).

Quando você usa o Amazon SNS para entregar mensagens de regiões de adesão para regiões habilitadas por padrão, é necessário alterar a política criada na função do AWS Lambda substituindo o principal `sns.amazonaws.com` por `sns.<opt-in-region>.amazonaws.com`.

Por exemplo, se você quiser inscrever uma função do Lambda no Leste dos EUA (Norte da Virgínia) em um tópico do SNS na Ásia-Pacífico (Hong Kong), altere o principal na política de função do AWS Lambda para `sns.ap-east-1.amazonaws.com`. As regiões de adesão incluem as regiões lançadas após 20 de março de 2019, que abrangem Ásia-Pacífico (Hong Kong), Oriente Médio (Bahrein), UE (Milão) e África (Cidade do Cabo). As regiões lançadas antes de 20 de março de 2019 são habilitadas por padrão.

#### Note

O AWS não oferece suporte à entrega entre regiões para o Lambda de uma região habilitada por padrão para uma região de adesão. Além disso, não há suporte para o encaminhamento de mensagens do SNS entre duas regiões de adesão.

## Inscrever uma função em um tópico

1. Faça login no [console do Amazon SNS](#).
2. No painel de navegação, escolha Topics (Tópicos).
3. Na página Topics (Tópicos), escolha um tópico.
4. Na seção Subscriptions (Inscrições), escolha Create subscription (Criar inscrição).
5. Na página Create subscription (Criar inscrição), na seção Details (detalhes), faça o seguinte:
  - a. Verifique o Topic ARN (ARN do tópico) escolhido.
  - b. Em Protocol (Protocolo), selecione AWS Lambda.
  - c. Em Endpoint, insira o ARN de uma função.
  - d. Selecione Create subscription.

Quando uma mensagem é publicada em um tópico do SNS que tem uma função do Lambda inscrita, a função do Lambda é chamada com a carga útil da mensagem publicada. Para obter informações

sobre como usar o AWS Lambda com o Amazon SNS, incluindo um tutorial, consulte [Usar o AWS Lambda com o Amazon SNS](#).

## Fanout para filas do Amazon SQS

O [Amazon SNS](#) funciona em conjunto com o Amazon Simple Queue Service (Amazon SQS). Esses serviços oferecem benefícios diferentes para os desenvolvedores. O Amazon SNS permite que aplicativos enviem mensagens urgentes para vários inscritos através de um mecanismo de “push”, eliminando a necessidade de verificar ou “pesquisar” periodicamente atualizações. O Amazon SQS é um serviço de fila de mensagens usado por aplicativos distribuídos para trocar mensagens por meio de um modelo de pesquisa e pode ser usado para decompor os componentes de envio e recebimento — sem a necessidade de cada componente estar disponível simultaneamente. Ao usar o Amazon SNS e o Amazon SQS em conjunto, as mensagens podem ser entregues a aplicativos que exigem uma notificação imediata de um evento e também podem continuar em uma fila do Amazon SQS para que outros aplicativos as processem posteriormente.

Ao inscrever uma fila do Amazon SQS em um tópico do Amazon SNS, é possível publicar uma mensagem no tópico e o Amazon SNS envia uma mensagem do Amazon SQS para a fila inscrita. A mensagem do Amazon SQS contém o assunto e a mensagem publicados no tópico juntamente com os metadados sobre a mensagem em um documento JSON. A mensagem do Amazon SQS será semelhante ao seguinte documento JSON.

```
{
  "Type" : "Notification",
  "MessageId" : "63a3f6b6-d533-4a47-aef9-fcf5cf758c76",
  "TopicArn" : "arn:aws:sns:us-west-2:123456789012:MyTopic",
  "Subject" : "Testing publish to subscribed queues",
  "Message" : "Hello world!",
  "Timestamp" : "2012-03-29T05:12:16.901Z",
  "SignatureVersion" : "1",
  "Signature" : "EXAMPLEnTrFPa3...",
  "SigningCertURL" : "https://sns.us-west-2.amazonaws.com/SimpleNotificationService-
f3ecfb7224c7233fe7bb5f59f96de52f.pem",
  "UnsubscribeURL" : "https://sns.us-west-2.amazonaws.com/?
Action=Unsubscribe&SubscriptionArn=arn:aws:sns:us-west-2:123456789012:MyTopic:c7fe3a54-
ab0e-4ec2-88e0-db410a0f2bee"
}
```

## Inscrever uma fila do Amazon SQS em um tópico do Amazon SNS

Para permitir que um tópico do Amazon SNS envie mensagens a uma fila do Amazon SQS, siga um destes procedimentos:

- Use o [console do Amazon SQS](#), o que simplifica o processo. Para obter mais informações, consulte [Subscribing an Amazon SQS queue to an Amazon SNS topic](#) no Guia do desenvolvedor do Amazon Simple Queue Service.
- Siga estas etapas:
  1. [Obtenha o Nome de recurso da Amazon \(ARN\) da fila para a qual deseja enviar mensagens e o tópico no qual você deseja inscrever a fila.](#)
  2. [Conceda a permissão sqs : SendMessage ao tópico do Amazon SNS para que ele possa enviar mensagens para a fila.](#)
  3. [Inscreva a fila no tópico do Amazon SNS.](#)
  4. [Forneça aos usuários do IAM ou às Contas da AWS as permissões apropriadas para publicar no tópico do Amazon SNS e ler mensagens da fila do Amazon SQS.](#)
  5. [Faça um teste publicando uma mensagem no tópico e lendo a mensagem a partir da fila.](#)

Para saber mais sobre como configurar um tópico para enviar mensagens a uma fila que está em uma conta da AWS diferente, consulte [Enviar mensagens do Amazon SNS para uma fila do SQS em uma conta diferente](#).

Para ver um modelo do AWS CloudFormation que cria um tópico que envia mensagens para duas filas, consulte [Usar um modelo do AWS CloudFormation para criar um tópico que envia mensagens para filas do Amazon SQS](#).

### Etapa 1: Obter o ARN da fila e do tópico

Ao inscrever uma fila em seu tópico, você precisará de uma cópia do ARN da fila. Da mesma forma, ao conceder permissão para o tópico enviar mensagens para a fila, você precisará de uma cópia do ARN do tópico.

Para obter o ARN da fila, você pode usar o console do Amazon SQS ou a ação de API [GetQueueAttributes](#).

Para obter o ARN da fila do console do Amazon SQS

1. Faça login no AWS Management Console e abra o console do Amazon SQS em <https://console.aws.amazon.com/sqs/>.
2. Marque a caixa de seleção da fila cujo ARN você deseja obter.
3. Na seção Details (Detalhes), copie o valor do ARN e use-o para se inscrever no tópico do Amazon SNS.

Para obter o ARN do tópico, use o console do Amazon SNS, o comando [sns-get-topic-attributes](#) ou a ação da API [GetQueueAttributes](#).

Para obter o ARN do tópico do console do Amazon SNS

1. Faça login no [console do Amazon SNS](#).
2. No painel de navegação, selecione o tópico do qual você deseja obter o ARN.
3. Na seção Details (Detalhes), copie o valor de ARN a fim de usá-lo para conceder permissão ao tópico do Amazon SNS para enviar mensagens à fila.

## Etapa 2: Conceder permissão ao tópico do Amazon SNS para enviar mensagens à fila do Amazon SQS

Para que um tópico do Amazon SNS possa de enviar mensagens a uma fila, você deve definir uma política na fila que permita que o tópico do Amazon SNS execute a ação `sqs:SendMessage`.

Antes de inscrever uma fila em um tópico, você precisa de um tópico e uma fila. Se ainda não criou um tópico ou uma fila, crie-os agora. Para obter mais informações, consulte [Criar um tópico](#) e [Criar uma fila](#) no Guia do desenvolvedor do Amazon Simple Queue Service.

Para definir uma política em uma fila, você pode usar o console do Amazon SQS ou a ação da API [SetQueueAttributes](#). Antes de começar, verifique se você tem o ARN do tópico que você deseja permitir que envie mensagens para a fila. Se você estiver assinando uma fila para vários tópicos, sua política deverá conter um elemento `Statement` para cada tópico.

Para definir uma política de `SendMessage` em uma fila usando o console do Amazon SQS

1. Faça login no AWS Management Console e abra o console do Amazon SQS em <https://console.aws.amazon.com/sqs/>.

2. Marque a caixa da fila cuja política você deseja definir, escolha a guia Access policy (Política de acesso) e escolha Edit (Editar).
3. No Política de acesso, defina quem pode acessar sua fila.
  - Adicione uma condição que permite a ação para o tópico.
  - Defina Principal como o serviço do Amazon SNS, conforme exibido no exemplo abaixo.
  - Use as chaves de condição globais [aws:SourceArn](#) ou [aws:SourceAccount](#) para se proteger contra o cenário [confused deputy](#). Para usar essas chaves de condição, defina o valor como o ARN de seu tópico. Se sua fila estiver inscrita em vários tópicos, use `aws:SourceAccount` em seu lugar.

Por exemplo, a política a seguir permite que MyTopic envie mensagens para MyQueue.

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "sns.amazonaws.com"
      },
      "Action": "sqs:SendMessage",
      "Resource": "arn:aws:sqs:us-east-2:123456789012:MyQueue",
      "Condition": {
        "ArnEquals": {
          "aws:SourceArn": "arn:aws:sns:us-east-2:123456789012:MyTopic"
        }
      }
    }
  ]
}
```

### Etapa 3: Inscrever a fila no tópico do Amazon SNS

Para enviar mensagens para uma fila por meio de um tópico, você deve inscrever a fila no tópico do Amazon SNS. Você especifica a fila por meio de sua ARN. Para se inscrever em um tópico, use o console do Amazon SNS, o comando [sns-subscribe](#) da CLI ou a ação da API [Subscribe](#). Antes de começar, verifique se você tem o ARN para a fila que deseja inscrever.

1. Faça login no [console do Amazon SNS](#).
2. No painel de navegação, escolha Topics (Tópicos).
3. Na página Topics (Tópicos), escolha um tópico.
4. Na página **MeuTópico**, na página Subscriptions (Inscrições), escolha Create subscription (Criar inscrição).
5. Na página Create subscription (Criar inscrição), na seção Details (detalhes), faça o seguinte:
  - a. Verifique o Topic ARN (ARN do tópico).
  - b. Em Protocol (Protocolo), escolha Amazon SQS.
  - c. Em Endpoint, insira o ARN de uma fila do Amazon SQS.
  - d. Escolha Create Subscription (Criar assinatura).

Quando a inscrição for confirmada, o ID da inscrição da nova inscrição exibirá o ID da inscrição. Se o proprietário da fila criar a inscrição, ela será automaticamente confirmada e deverá ser ativada quase que imediatamente.

Geralmente, você inscreverá sua própria fila em seu próprio tópico em sua própria conta. No entanto, você também pode inscrever uma fila de uma conta diferente em seu tópico. Se o usuário que criar a inscrição não for o proprietário da fila (por exemplo, se um usuário da conta A inscreve uma fila da conta B para um tópico na conta A), a inscrição deverá ser confirmada. Para obter mais informações sobre como inscrever uma fila de uma conta diferente e confirmar a inscrição, consulte [Enviar mensagens do Amazon SNS para uma fila do SQS em uma conta diferente](#).

#### Etapa 4: Conceder aos usuários permissões para o tópico apropriado e as ações da fila

Você deve usar o AWS Identity and Access Management (IAM) para permitir que apenas usuários apropriados publiquem no tópico do Amazon SNS e leiam/excluam mensagens da fila do Amazon SQS. Para obter mais informações sobre como controlar ações em tópicos e filas para usuários do IAM, consulte [Usar políticas baseadas em identidade com o Amazon SNS](#), [Identity and Access Management no Amazon SQS](#) no Guia do desenvolvedor do Amazon Simple Queue Service.

Há duas maneiras de controlar o acesso a um tópico ou uma fila:

- [Adicione uma política para um usuário ou grupo do IAM](#). A maneira mais simples de conceder aos usuários permissões para tópicos ou filas é criar um grupo e adicionar a política adequada e os usuários ao grupo. É muito mais fácil adicionar e remover usuários de um grupo do que controlar as políticas definidas para usuários individualmente.
- [Adicione uma política ao tópico ou à fila](#). Se deseja conceder permissões para um tópico ou fila para outra conta da AWS, a única maneira de fazer isso é adicionando uma política com a Conta da AWS para a qual você deseja conceder permissões como principal.

Você deve usar o primeiro método para a maioria dos casos (aplicar políticas a grupos e gerenciar permissões para usuários adicionando ou removendo os usuários adequados aos grupos). Se você precisa conceder permissões para um usuário em outra conta, use o segundo método.

### Adicionar uma política a um usuário ou grupo do IAM

Se você adicionou a política a seguir a um usuário ou grupo do IAM estará concedendo a esse usuário ou membros desse grupo permissão para executar a ação `sns:Publish` no tópico `MyTopic`.

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "sns:Publish",
      "Resource": "arn:aws:sns:us-east-2:123456789012:MyTopic"
    }
  ]
}
```

Se você tiver adicionado a política a seguir a um usuário ou grupo do IAM, estará concedendo a esse usuário ou membros do grupo permissão para executar as ações `sqs:ReceiveMessage` e `sqs:DeleteMessage` nas filas `MyQueue1` e `MyQueue2`.

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "sqs:ReceiveMessage",
        "sqs:DeleteMessage"
      ],
      "Resource": [
```

```
        "arn:aws:sqs:us-east-2:123456789012:MyQueue1",
        "arn:aws:sqs:us-east-2:123456789012:MyQueue2"
    ]
}
]
```

## Adição de uma política a um tópico ou uma fila

Os exemplos de políticas a seguir mostram como conceder a outra conta permissões para um tópico e uma fila.

### Note

Ao conceder à outra Conta da AWS acesso a um recurso em sua conta, você também está concedendo permissões para o recurso em questão aos usuários do IAM com acesso de nível de administrador (acesso coringa). Todos os outros usuários do IAM na outra conta têm automaticamente o acesso negado a seus recursos. Se desejar conceder a usuários específicos do IAM dessa Conta da AWS acesso a seu recurso, a conta ou um usuário do IAM com acesso de nível de administrador deverá delegar permissões para o recurso para esses usuários do IAM. Para obter mais informações sobre a delegação entre contas, consulte [Habilitar acesso entre contas](#) no Guia de uso do IAM.

Se adicionou a seguinte política a um tópico MeuTópico na conta, você pode conceder à conta 123456789012 permissão 111122223333 para executar a ação `sns:Publish` no tópico.

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "111122223333"
      },
      "Action": "sns:Publish",
      "Resource": "arn:aws:sns:us-east-2:123456789012:MyTopic"
    }
  ]
}
```

Se adicionar a seguinte política à fila MyQueue na conta 123456789012, você concederá à conta 111122223333 permissão para executar as ações `sqs:ReceiveMessage` e `sqs:DeleteMessage` na fila.

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "111122223333"
      },
      "Action": [
        "sqs:DeleteMessage",
        "sqs:ReceiveMessage"
      ],
      "Resource": [
        "arn:aws:sqs:us-east-2:123456789012:MyQueue"
      ]
    }
  ]
}
```

## Etapa 5: Testar as assinaturas de fila do tópico

Você pode testar as inscrições de fila de um tópico publicando nele e visualizando a mensagem que ele envia para a fila.

Para publicar um tópico usando o console do Amazon SNS

1. Usando as credenciais da Conta da AWS ou do usuário do IAM com permissão para publicar no tópico, faça login no AWS Management Console e abra o console do Amazon SNS em <https://console.aws.amazon.com/sns/>.
2. No painel de navegação, escolha o tópico e selecione Publish to Topic (Publicar no tópico).
3. Na caixa Subject (Assunto), digite um assunto (por exemplo, **Testing publish to queue**), na caixa Message (Mensagem), digite algum texto (por exemplo, **Hello world!**) e escolha Publish Message (Publicar mensagem). Será exibida a seguinte mensagem: Your message has been successfully published.

## Para visualizar a mensagem do tópico usando o console do Amazon SQS

1. Usando as credenciais da Conta da AWS ou do usuário do IAM com permissão para visualizar mensagens na fila, faça login no AWS Management Console e abra o console do Amazon SQS em <https://console.aws.amazon.com/sqs/>.
2. Escolha uma queue (fila) que esteja inscrita no tópico.
3. Escolha Send and receive messages (Enviar e receber mensagens), depois escolha Poll for messages (Pesquisar mensagens). Uma mensagem com um tipo de Notificação é exibida.
4. Na coluna Body (Corpo), escolha More Details (Mais detalhes). A caixa Detalhes da mensagem inclui um documento JSON que contém o assunto e a mensagem publicada no tópico. A mensagem é semelhante ao seguinte documento JSON.

```
{
  "Type" : "Notification",
  "MessageId" : "63a3f6b6-d533-4a47-aef9-fcf5cf758c76",
  "TopicArn" : "arn:aws:sns:us-west-2:123456789012:MyTopic",
  "Subject" : "Testing publish to subscribed queues",
  "Message" : "Hello world!",
  "Timestamp" : "2012-03-29T05:12:16.901Z",
  "SignatureVersion" : "1",
  "Signature" : "EXAMPLEnTrFPa3...",
  "SigningCertURL" : "https://sns.us-west-2.amazonaws.com/
SimpleNotificationService-f3ecfb7224c7233fe7bb5f59f96de52f.pem",
  "UnsubscribeURL" : "https://sns.us-west-2.amazonaws.com/?
Action=Unsubscribe&SubscriptionArn=arn:aws:sns:us-
west-2:123456789012:MyTopic:c7fe3a54-ab0e-4ec2-88e0-db410a0f2bee"
}
```

5. Escolha Close (Fechar). Você publicou com sucesso em um tópico que envia mensagens de notificação para uma fila.

## Usar um modelo do AWS CloudFormation para criar um tópico que envia mensagens para filas do Amazon SQS

O AWS CloudFormation permite que você use um arquivo de modelo para criar e configurar um conjunto de recursos da AWS como uma unidade única. Esta seção tem um modelo de exemplo que facilita a implantação de tópicos que publicam em filas. Os modelos processam as etapas de configuração para você, criando duas filas, criando um tópico com inscrições das filas, adicionando

uma política às filas para que o tópico possa enviar mensagens para as filas e criando usuários e grupos do IAM para controlar o acesso a esses recursos.

Para obter mais informações sobre a implantação de recursos da AWS usando um modelo do AWS CloudFormation, consulte [Get Started](#) (“Conceitos básicos”) no Manual do usuário do AWS CloudFormation.

## Uso de um modelo do AWS CloudFormation para definir tópicos e filas dentro de uma Conta da AWS

O modelo de exemplo cria um tópico do Amazon SNS que pode enviar mensagens para duas filas do Amazon SQS com as permissões adequadas para que os membros de um grupo do IAM publiquem no tópico e os membros do outro leiam mensagens das filas. O modelo também cria usuários do IAM que são adicionados a cada grupo.

Copie o conteúdo do modelo em um arquivo. Também é possível fazer download do modelo na [página Modelos do AWS CloudFormation](#). Na página de modelos, selecione Procurar modelos de exemplo por AWS Serviço da e, depois, escolha Amazon Simple Queue Service.

MySNSTopic está configurado para publicar dois endpoints inscritos, que são duas filas do Amazon SQS (MyQueue1 e MyQueue2). MyPublishTopicGroup é um grupo do IAM cujos membros têm permissão para publicar no MySNSTopic usando a ação da API [Publish](#) ou o comando [sns-publish](#). O modelo cria os usuários do IAM MyQueueUser e MyPublishUser e oferece a eles perfis de login e chaves de acesso. O usuário que cria uma pilha com esse modelo especifica as senhas para os perfis de login como parâmetros de entrada. O modelo cria chaves de acesso para os dois usuários do IAM com MyQueueUserKey e MyPublishUserKey. AddUserToMyPublishTopicGroup adiciona MyPublishUser ao MyPublishTopicGroup para que o usuário tenha as permissões atribuídas ao grupo.

MyRDMessageQueueGroup é um grupo do IAM cujos membros têm permissão para ler e excluir mensagens das duas filas do Amazon SQS usando as ações da API [ReceiveMessage](#) e [DeleteMessage](#). AddUserToMyQueueGroup adiciona MyQueueUser ao MyRDMessageQueueGroup para que o usuário tenha as permissões atribuídas ao grupo. MyQueuePolicy concede permissão para MySNSTopic publicar suas notificações nas duas filas.

A listagem a seguir mostra o conteúdo de modelos do AWS CloudFormation.

```
{
  "AWSTemplateFormatVersion" : "2010-09-09",
```

```
"Description" : "AWS CloudFormation Sample Template SNSToSQS: This Template creates
an SNS topic that can send messages to
two SQS queues with appropriate permissions for one IAM user to publish to the topic
and another to read messages from the queues.
MySNSTopic is set up to publish to two subscribed endpoints, which are two SQS queues
(MyQueue1 and MyQueue2). MyPublishUser is an IAM user
that can publish to MySNSTopic using the Publish API. MyTopicPolicy assigns that
permission to MyPublishUser. MyQueueUser is an IAM user
that can read messages from the two SQS queues. MyQueuePolicy assigns those
permissions to MyQueueUser. It also assigns permission for
MySNSTopic to publish its notifications to the two queues. The template creates
access keys for the two IAM users with MyPublishUserKey
and MyQueueUserKey. ***Warning*** you will be billed for the AWS resources used if
you create a stack from this template.",
```

```
"Parameters": {
  "MyPublishUserPassword": {
    "NoEcho": "true",
    "Type": "String",
    "Description": "Password for the IAM user MyPublishUser",
    "MinLength": "1",
    "MaxLength": "41",
    "AllowedPattern": "[a-zA-Z0-9]*",
    "ConstraintDescription": "must contain only alphanumeric characters."
  },
  "MyQueueUserPassword": {
    "NoEcho": "true",
    "Type": "String",
    "Description": "Password for the IAM user MyQueueUser",
    "MinLength": "1",
    "MaxLength": "41",
    "AllowedPattern": "[a-zA-Z0-9]*",
    "ConstraintDescription": "must contain only alphanumeric characters."
  }
},
```

```
"Resources": {
  "MySNSTopic": {
    "Type": "AWS::SNS::Topic",
    "Properties": {
      "Subscription": [{
        "Endpoint": {
          "Fn::GetAtt": ["MyQueue1", "Arn"]
        }
      }
    ]
  }
}
```

```

        },
        "Protocol": "sqs"
    },
    {
        "Endpoint": {
            "Fn::GetAtt": ["MyQueue2", "Arn"]
        },
        "Protocol": "sqs"
    }
]
}
},
"MyQueue1": {
    "Type": "AWS::SQS::Queue"
},
"MyQueue2": {
    "Type": "AWS::SQS::Queue"
},
"MyPublishUser": {
    "Type": "AWS::IAM::User",
    "Properties": {
        "LoginProfile": {
            "Password": {
                "Ref": "MyPublishUserPassword"
            }
        }
    }
},
"MyPublishUserKey": {
    "Type": "AWS::IAM::AccessKey",
    "Properties": {
        "UserName": {
            "Ref": "MyPublishUser"
        }
    }
},
"MyPublishTopicGroup": {
    "Type": "AWS::IAM::Group",
    "Properties": {
        "Policies": [{
            "PolicyName": "MyTopicGroupPolicy",
            "PolicyDocument": {
                "Statement": [{
                    "Effect": "Allow",

```

```
        "Action": [
            "sns:Publish"
        ],
        "Resource": {
            "Ref": "MySNSTopic"
        }
    }
}]]
}
]]
}
},
"AddUserToMyPublishTopicGroup": {
    "Type": "AWS::IAM::UserToGroupAddition",
    "Properties": {
        "GroupName": {
            "Ref": "MyPublishTopicGroup"
        },
        "Users": [{
            "Ref": "MyPublishUser"
        }]
    }
},
"MyQueueUser": {
    "Type": "AWS::IAM::User",
    "Properties": {
        "LoginProfile": {
            "Password": {
                "Ref": "MyQueueUserPassword"
            }
        }
    }
},
"MyQueueUserKey": {
    "Type": "AWS::IAM::AccessKey",
    "Properties": {
        "UserName": {
            "Ref": "MyQueueUser"
        }
    }
},
"MyRDMessageQueueGroup": {
    "Type": "AWS::IAM::Group",
    "Properties": {
        "Policies": [{
```

```

    "PolicyName": "MyQueueGroupPolicy",
    "PolicyDocument": {
      "Statement": [{
        "Effect": "Allow",
        "Action": [
          "sqs:DeleteMessage",
          "sqs:ReceiveMessage"
        ],
        "Resource": [{
          "Fn::GetAtt": ["MyQueue1", "Arn"]
        },
        {
          "Fn::GetAtt": ["MyQueue2", "Arn"]
        }
      ]
    }
  ]
}
}],
}
},
"AddUserToMyQueueGroup": {
  "Type": "AWS::IAM::UserToGroupAddition",
  "Properties": {
    "GroupName": {
      "Ref": "MyRDMessageQueueGroup"
    },
    "Users": [{
      "Ref": "MyQueueUser"
    }
  ]
}
},
"MyQueuePolicy": {
  "Type": "AWS::SQS::QueuePolicy",
  "Properties": {
    "PolicyDocument": {
      "Statement": [{
        "Effect": "Allow",
        "Principal": {
          "Service": "sns.amazonaws.com"
        },
        "Action": ["sqs:SendMessage"],
        "Resource": "*",
        "Condition": {
          "ArnEquals": {

```

```
        "aws:SourceArn": {
            "Ref": "MySNSTopic"
        }
    }
}
}],
},
"Queues": [{
    "Ref": "MyQueue1"
}, {
    "Ref": "MyQueue2"
}]
}
}
},
"Outputs": {
    "MySNSTopicTopicARN": {
        "Value": {
            "Ref": "MySNSTopic"
        }
    },
    "MyQueue1Info": {
        "Value": {
            "Fn::Join": [
                " ",
                [
                    "ARN:",
                    {
                        "Fn::GetAtt": ["MyQueue1", "Arn"]
                    },
                    "URL:",
                    {
                        "Ref": "MyQueue1"
                    }
                ]
            ]
        }
    },
    "MyQueue2Info": {
        "Value": {
            "Fn::Join": [
                " ",
                [
                    "ARN:",
```

```
        {
          "Fn::GetAtt": ["MyQueue2", "Arn"]
        },
        "URL:",
        {
          "Ref": "MyQueue2"
        }
      ]
    ]
  },
  "MyPublishUserInfo": {
    "Value": {
      "Fn::Join": [
        " ",
        [
          "ARN:",
          {
            "Fn::GetAtt": ["MyPublishUser", "Arn"]
          },
          "Access Key:",
          {
            "Ref": "MyPublishUserKey"
          },
          "Secret Key:",
          {
            "Fn::GetAtt": ["MyPublishUserKey", "SecretAccessKey"]
          }
        ]
      ]
    }
  },
  "MyQueueUserInfo": {
    "Value": {
      "Fn::Join": [
        " ",
        [
          "ARN:",
          {
            "Fn::GetAtt": ["MyQueueUser", "Arn"]
          },
          "Access Key:",
          {
            "Ref": "MyQueueUserKey"
          }
        ]
      ]
    }
  }
}
```

```
    },
    "Secret Key:",
    {
      "Fn::GetAtt": ["MyQueueUserKey", "SecretAccessKey"]
    }
  ]
]
}
}
```

## Fanout para endpoints HTTP(S)

Use o [Amazon SNS](#) para enviar mensagens de notificação a um ou mais endpoints HTTP ou HTTPS. Quando você inscrever um endpoint em um tópico, será possível publicar uma notificação no tópico, e o Amazon SNS envia uma solicitação HTTP POST entregando o conteúdo da notificação para o endpoint inscrito. Ao inscrever o endpoint, você seleciona se o Amazon SNS usa HTTP ou HTTPS para enviar a solicitação POST ao endpoint. Se você usar HTTPS, poderá aproveitar o suporte do Amazon SNS para o seguinte:

- **Server Name Indication (SNI):** permite que o Amazon SNS ofereça suporte a endpoints HTTPS que exigem SNI, como um servidor que exige vários certificados para hospedar vários domínios. Para obter mais informações sobre SNI, consulte [Server Name Indication](#).
- **Autenticação de acesso Basic e Digest:** permite que você especifique um nome de usuário e uma senha no URL HTTPS para a solicitação HTTP POST, como `https://user:password@domain.com` ou `https://user@domain.com`. O nome do usuário e a senha são criptografados pela conexão SSL estabelecida ao usar HTTPS. Somente o nome de domínio é enviado em texto sem formatação. Para obter mais informações sobre autenticação e controle de acesso Basic e Digest, consulte a [RFC-2617](#).

### Important

No momento, o Amazon SNS não é compatível com endpoints HTTP(S) privados. URLs HTTPS só podem ser recuperados a partir da ação da API `GetSubscriptionAttributes` do Amazon SNS, para as entidades principais às quais você concedeu acesso à API.

**Note**

O serviço de cliente deve poder oferecer suporte à resposta de cabeçalho HTTP/1.1 401 Unauthorized

A solicitação contém o assunto e a mensagem que foram publicados no tópico juntamente com os metadados sobre a notificação em um documento JSON. A solicitação será semelhante à seguinte solicitação POST HTTP. Para obter detalhes sobre o cabeçalho HTTP e o formato JSON do corpo de solicitação, consulte [Cabeçalhos HTTP/HTTPS](#) e [Formato JSON de notificação HTTP/HTTPS](#).

```
POST / HTTP/1.1
```

```
x-amz-sns-message-type: Notification
x-amz-sns-message-id: da41e39f-ea4d-435a-b922-c6aae3915ebe
x-amz-sns-topic-arn: arn:aws:sns:us-west-2:123456789012:MyTopic
x-amz-sns-subscription-arn: arn:aws:sns:us-
west-2:123456789012:MyTopic:2bcfbf39-05c3-41de-beaa-fcfcc21c8f55
Content-Length: 761
Content-Type: text/plain; charset=UTF-8
Host: ec2-50-17-44-49.compute-1.amazonaws.com
Connection: Keep-Alive
User-Agent: Amazon Simple Notification Service Agent
```

```
{
  "Type" : "Notification",
  "MessageId" : "da41e39f-ea4d-435a-b922-c6aae3915ebe",
  "TopicArn" : "arn:aws:sns:us-west-2:123456789012:MyTopic",
  "Subject" : "test",
  "Message" : "test message",
  "Timestamp" : "2012-04-25T21:49:25.719Z",
  "SignatureVersion" : "1",
  "Signature" :
  "EXAMPLE1DMXvB8r9R83tGoNn0ecwd5UjllzsvSvbItzfaMpN2nk5HVSsw7Xn0n/49IkxDKz8Yr1H2qJXj2iZB0Zo2071c4
  "SigningCertURL" : "https://sns.us-west-2.amazonaws.com/SimpleNotificationService-
f3ecfb7224c7233fe7bb5f59f96de52f.pem",
  "UnsubscribeURL" : "https://sns.us-west-2.amazonaws.com/?
Action=Unsubscribe&SubscriptionArn=arn:aws:sns:us-
west-2:123456789012:MyTopic:2bcfbf39-05c3-41de-beaa-fcfcc21c8f55"
}
```

## Tópicos

- [Inscrever um endpoint HTTP/S em um tópico](#)
- [Verificação das assinaturas de mensagens do Amazon SNS](#)
- [Analisar formatos de mensagens](#)

## Inscrever um endpoint HTTP/S em um tópico

As páginas desta seção descrevem como inscrever endpoints HTTP/S nos tópicos do Amazon SNS.

### Tópicos

- [Etapa 1: Verificar se o endpoint está pronto para processar mensagens do Amazon SNS](#)
- [Etapa 2: Inscrever o endpoint HTTP/HTTPS no tópico do Amazon SNS](#)
- [Etapa 3: Confirmar a inscrição](#)
- [Etapa 4: definir a política de entrega para a assinatura \(opcional\)](#)
- [Etapa 5: Conceder aos usuários permissões para publicar no tópico \(opcional\)](#)
- [Etapa 6: Enviar mensagens para o endpoint HTTP/HTTPS](#)

### Etapa 1: Verificar se o endpoint está pronto para processar mensagens do Amazon SNS

Antes de inscrever um endpoint HTTP ou HTTPS em um tópico, você deve garantir que ele tem a capacidade de lidar com as solicitações HTTP POST que o Amazon SNS usa para enviar a confirmação de inscrição e as mensagens de notificação. Geralmente, isso significa a criação e a implantação de uma aplicação web (por exemplo, um servlet Java se o host do endpoint está executando Linux com Apache e Tomcat) que processa solicitações HTTP do Amazon SNS. Quando você inscreve um endpoint HTTP, o Amazon SNS envia uma solicitação de confirmação de inscrição. Seu endpoint deve estar preparado para receber e processar essa solicitação quando você criar a assinatura, porque o Amazon SNS envia essa solicitação nesse momento. O Amazon SNS não enviará notificações ao endpoint até que a assinatura seja confirmada. Assim que você confirmar a inscrição, o Amazon SNS enviará notificações para o endpoint quando uma ação de publicação for realizada no tópico inscrito.

Para configurar o endpoint para processar as mensagens de notificação e de confirmação de inscrição

1. Seu código deve conter os cabeçalhos HTTP de solicitações HTTP POST que o Amazon SNS envia para seu endpoint. O código deve procurar o campo do cabeçalho `x-amz-sns-message-type`, que mostra o tipo de mensagem que o Amazon SNS enviou a você. Ao visualizar o cabeçalho, você pode determinar o tipo da mensagem sem a necessidade de analisar o corpo da solicitação HTTP. Há dois tipos com os quais você precisa lidar: `SubscriptionConfirmation` e `Notification`. A mensagem `UnsubscribeConfirmation` é usada somente quando a assinatura é excluída do tópico.

Para obter detalhes sobre o cabeçalho HTTP, consulte [Cabeçalhos HTTP/HTTPS](#). A seguinte solicitação HTTP POST é um exemplo de uma mensagem de confirmação de inscrição.

```
POST / HTTP/1.1
x-amz-sns-message-type: SubscriptionConfirmation
x-amz-sns-message-id: 165545c9-2a5c-472c-8df2-7ff2be2b3b1b
x-amz-sns-topic-arn: arn:aws:sns:us-west-2:123456789012:MyTopic
Content-Length: 1336
Content-Type: text/plain; charset=UTF-8
Host: example.com
Connection: Keep-Alive
User-Agent: Amazon Simple Notification Service Agent

{
  "Type" : "SubscriptionConfirmation",
  "MessageId" : "165545c9-2a5c-472c-8df2-7ff2be2b3b1b",
  "Token" : "2336412f37f...",
  "TopicArn" : "arn:aws:sns:us-west-2:123456789012:MyTopic",
  "Message" : "You have chosen to subscribe to the topic arn:aws:sns:us-
west-2:123456789012:MyTopic.\nTo confirm the subscription, visit the SubscribeURL
included in this message.",
  "SubscribeURL" : "https://sns.us-west-2.amazonaws.com/?
Action=ConfirmSubscription&TopicArn=arn:aws:sns:us-
west-2:123456789012:MyTopic&Token=2336412f37...",
  "Timestamp" : "2012-04-26T20:45:04.751Z",
  "SignatureVersion" : "1",
  "Signature" : "EXAMPLEpH+...",
  "SigningCertURL" : "https://sns.us-west-2.amazonaws.com/
SimpleNotificationService-f3ecfb7224c7233fe7bb5f59f96de52f.pem"
}
```

2. Seu código deve analisar o documento JSON no corpo da solicitação HTTP POST e o tipo de conteúdo texto/sem formatação para ler os pares de nome/valor que compõem a mensagem do Amazon SNS. Use um analisador JSON que faça a conversão da representação em sequência de escape dos caracteres de controle de volta para seus valores de caracteres ASCII (por exemplo, converter `\n` para um caractere de nova linha). Você pode usar um analisador JSON existente, como o [Processador Jackson JSON](#) ou escrever seu próprio analisador. Para enviar o texto nos campos de assunto e mensagem em formato JSON válido, o Amazon SNS deve converter alguns caracteres de controle para representações em sequência de escape para inclusão no documento JSON. Ao receber o documento JSON no corpo da solicitação POST enviada para o endpoint, você deve converter os caracteres em sequência de escape de volta para seus valores de caractere originais se quiser uma representação exata do assunto e das mensagens originais publicadas no tópico. Isso é essencial se você deseja verificar a assinatura de uma notificação, pois a assinatura usa a mensagem e o assunto em seus formulários originais como parte da string para assinar.
3. Seu código deve verificar a autenticidade de uma mensagem de notificação, confirmação de inscrição ou confirmação de cancelamento de inscrição enviada pelo Amazon SNS. O uso das informações contidas na mensagem do Amazon SNS, o endpoint pode recriar a assinatura para que você possa verificar o conteúdo da mensagem comparando sua assinatura com a assinatura enviada pelo Amazon SNS junto com a mensagem. Para obter mais informações sobre como verificar a assinatura de uma mensagem, consulte [Verificação das assinaturas de mensagens do Amazon SNS](#).
4. Com base no tipo especificado pelo campo de cabeçalho `x-amz-sns-message-type`, seu código deve ler o documento JSON contido no corpo da solicitação HTTP e processar a mensagem. A seguir, são apresentadas as diretrizes para lidar com os dois tipos principais de mensagens:

### SubscriptionConfirmation

Leia o valor para `SubscribeURL` e acesse o URL. Para confirmar a assinatura e começar a receber notificações no endpoint, é necessário visitar o `URLSubscribeURL` (por exemplo, enviando uma solicitação HTTP GET para o URL). Consulte o exemplo de solicitação HTTP na etapa anterior para ver qual a aparência de `SubscribeURL`. Para obter mais informações sobre o formato da mensagem `SubscriptionConfirmation`, consulte [Formato JSON de confirmação de assinatura HTTP/HTTPS](#). Ao acessar o URL, você receberá de volta uma resposta que se parece com o seguinte documento XML. O documento retorna o ARN da inscrição para o endpoint no elemento `ConfirmSubscriptionResult`.

```
<ConfirmSubscriptionResponse xmlns="http://sns.amazonaws.com/doc/2010-03-31/">
  <ConfirmSubscriptionResult>
    <SubscriptionArn>arn:aws:sns:us-
west-2:123456789012:MyTopic:2bcfbf39-05c3-41de-beaa-fcfcc21c8f55</
SubscriptionArn>
  </ConfirmSubscriptionResult>
  <ResponseMetadata>
    <RequestId>075ecce8-8dac-11e1-bf80-f781d96e9307</RequestId>
  </ResponseMetadata>
</ConfirmSubscriptionResponse>
```

Como alternativa para visitar o `SubscribeURL`, você pode confirmar a inscrição usando a ação [ConfirmSubscription](#) com o Token definido como seu valor correspondente na mensagem `SubscriptionConfirmation`. Se você deseja permitir que somente o proprietário do tópico e o proprietário da inscrição cancelem o recebimento para o endpoint, chame a ação `ConfirmSubscription` com uma assinatura AWS.

## Notificação

Leia os valores de `Subject` e `Message` para obter as informações da notificação que foi publicada no tópico.

Para obter mais detalhes sobre o formato da mensagem `Notification`, consulte [Cabeçalhos HTTP/HTTPS](#). A seguinte solicitação HTTP POST é um exemplo de uma mensagem de notificação enviada para o endpoint `example.com`.

```
POST / HTTP/1.1
x-amz-sns-message-type: Notification
x-amz-sns-message-id: 22b80b92-fdea-4c2c-8f9d-bdfb0c7bf324
x-amz-sns-topic-arn: arn:aws:sns:us-west-2:123456789012:MyTopic
x-amz-sns-subscription-arn: arn:aws:sns:us-
west-2:123456789012:MyTopic:c9135db0-26c4-47ec-8998-413945fb5a96
Content-Length: 773
Content-Type: text/plain; charset=UTF-8
Host: example.com
Connection: Keep-Alive
User-Agent: Amazon Simple Notification Service Agent

{
  "Type" : "Notification",
  "MessageId" : "22b80b92-fdea-4c2c-8f9d-bdfb0c7bf324",
```

```
"TopicArn" : "arn:aws:sns:us-west-2:123456789012:MyTopic",
"Subject" : "My First Message",
"Message" : "Hello world!",
"Timestamp" : "2012-05-02T00:54:06.655Z",
"SignatureVersion" : "1",
"Signature" : "EXAMPLEw6JRN...",
"SigningCertURL" : "https://sns.us-west-2.amazonaws.com/
SimpleNotificationService-f3ecfb7224c7233fe7bb5f59f96de52f.pem",
"UnsubscribeURL" : "https://sns.us-west-2.amazonaws.com/?
Action=Unsubscribe&SubscriptionArn=arn:aws:sns:us-
west-2:123456789012:MyTopic:c9135db0-26c4-47ec-8998-413945fb5a96"
}
```

5. Certifique-se de que o endpoint responde à mensagem de HTTP POST do Amazon SNS com o código de status apropriado. A conexão atingirá o tempo limite em 15 segundos. Se o endpoint não responder antes da conexão atingir o tempo limite ou retornar um código de status fora do intervalo de 200–4xx, o Amazon SNS considerará que houve falha na entrega da mensagem.
6. Certifique-se de que o seu código pode lidar com novas tentativas de entrega de mensagens do Amazon SNS. Se o Amazon SNS não receber uma resposta bem-sucedida do endpoint, ele tenta entregar a mensagem novamente. Isso se aplica a todas as mensagens, incluindo a mensagem de confirmação de inscrição. Por padrão, se a entrega inicial da mensagem falhar, o Amazon SNS faz até três novas tentativas com uma espera de 20 segundos entre as tentativas malsucedidas.

#### Note

A solicitação da mensagem atinge o tempo limite em 15 segundos. Isso significa que se a falha na entrega da mensagem for causada por exceder o tempo limite, o Amazon SNS tentará outra vez por aproximadamente 35 segundos após a última tentativa de entrega. Você pode definir uma política de entrega diferente para o endpoint.

O Amazon SNS usa o campo de cabeçalho `x-amz-sns-message-id` para identificar de forma exclusiva cada mensagem publicada em um tópico do Amazon SNS. Ao comparar os IDs das mensagens que foram processadas com as mensagens recebidas, você pode determinar se a mensagem é uma nova tentativa.

7. Se estiver inscrevendo um endpoint HTTPS, verifique se o endpoint tem um certificado do servidor de uma autoridade de certificado (CA) confiável. O Amazon SNS só envia mensagens

a endpoints HTTPS que têm um certificado de servidor assinado por uma CA confiável para o Amazon SNS.

8. Implante o código que você criou para receber mensagens do Amazon SNS. Quando você assinar o endpoint, ele deve estar pronto para receber pelo menos a mensagem de confirmação de inscrição.

## Etapa 2: Inscrever o endpoint HTTP/HTTPS no tópico do Amazon SNS

Para enviar mensagens a um endpoint HTTP ou HTTPS por meio de um tópico, inscreva o endpoint no tópico do Amazon SNS. Você especifica o endpoint usando seu URL. Para inscrever-se em um tópico, você pode usar o console do Amazon SNS, o comando [sns-subscribe](#) ou a ação da API [Subscribe](#). Antes de começar, verifique se você tem o URL do endpoint que deseja inscrever e se o endpoint está preparado para receber a confirmação e as mensagens de notificação, como descrito na Etapa 1.

Para inscrever um endpoint HTTP ou HTTPS em um tópico usando o console do Amazon SNS

1. Faça login no [console do Amazon SNS](#).
2. No painel de navegação, escolha Topics (Tópicos).
3. Selecione Create subscription (Criar assinatura).
4. Na lista suspensa Protocolo, selecione HTTP ou HTTPS.
5. Na caixa Endpoint, cole a URL do endpoint para o qual você deseja que o tópico envie mensagens e escolha Create subscription (Criar inscrição).
6. A mensagem de confirmação é exibida. Escolha Close (Fechar).

Seu novo ID da inscrição exibe PendingConfirmation. Quando você confirmar a inscrição, o ID da inscrição exibirá o ID da inscrição.

## Etapa 3: Confirmar a inscrição

Depois que você se inscrever em um endpoint, o Amazon SNS enviará uma mensagem de confirmação de assinatura ao endpoint. Você já deve ter um código que executa as ações descritas na [Etapa 1](#) implantado no endpoint. Especificamente, o código no endpoint deve recuperar o valor `SubscribeURL` na mensagem de confirmação de assinatura e acessar o local especificado pelo próprio `SubscribeURL` ou disponibilizá-lo para que você possa acessar o `SubscribeURL` manualmente (por exemplo, se você estiver usando um navegador da Web). O Amazon SNS

não enviará mensagens ao endpoint até que a assinatura tenha sido confirmada. Ao acessar o `SubscribeURL`, a resposta conterá um documento XML com um elemento `SubscriptionArn` que especifica o ARN da inscrição. Você também pode usar o console do Amazon SNS para verificar se a inscrição foi confirmada: o `Subscription ID` (ID da inscrição) exibirá o ARN da inscrição, em vez do valor `PendingConfirmation` exibido quando você adicionou a inscrição.

#### Etapa 4: definir a política de entrega para a assinatura (opcional)

Por padrão, se a entrega inicial da mensagem falhar, o Amazon SNS faz até três novas tentativas com uma espera de 20 segundos entre as tentativas malsucedidas. Conforme foi dito na [Etapa 1](#), o endpoint deve ter o código que pode processar novas tentativas de mensagens. Ao definir a política de entrega em um tópico ou inscrição, você pode controlar a frequência e o intervalo com que o Amazon SNS tentará entregar novamente mensagens com falha. Você também pode especificar o tipo de conteúdo para as notificações HTTP/S em `DeliveryPolicy`. Para obter mais informações, consulte [Como criar uma política de entrega HTTP/S](#).

#### Etapa 5: Conceder aos usuários permissões para publicar no tópico (opcional)

Por padrão, o proprietário do tópico tem permissão para publicar no tópico. Para permitir que outros usuários ou aplicativos publiquem no tópico, você deve usar o AWS Identity and Access Management (IAM) para conceder permissão pública ao tópico. Para obter mais informações sobre como conceder permissões para ações do Amazon SNS a usuários do IAM, consulte [Usar políticas baseadas em identidade com o Amazon SNS](#).

Há duas maneiras de controlar o acesso a um tópico:

- Adicione uma política para um usuário ou grupo do IAM. A maneira mais simples de conceder aos usuários permissões para tópicos é criar um grupo e adicionar a política adequada e os usuários ao grupo. É muito mais fácil adicionar e remover usuários de um grupo do que controlar as políticas definidas para usuários individualmente.
- Adicione uma política ao tópico. Se deseja conceder permissões para um tópico para outra AWS, a única maneira de fazer isso é adicionando uma política com a Conta da AWS para a qual você deseja conceder permissões como principal.

Você deve usar o primeiro método para a maioria dos casos (aplicar políticas a grupos e gerenciar permissões para usuários adicionando ou removendo os usuários adequados aos grupos). Se você precisa conceder permissões a um usuário em outra conta, use o segundo método.

Se você tiver adicionado a política a seguir a um usuário ou grupo do IAM estará concedendo a esse usuário ou membros desse grupo permissão para executar a ação `sns:Publish` no tópico `MyTopic`.

```
{
  "Statement": [{
    "Sid": "AllowPublishToMyTopic",
    "Effect": "Allow",
    "Action": "sns:Publish",
    "Resource": "arn:aws:sns:us-east-2:123456789012:MyTopic"
  }]
}
```

O exemplo a seguir de política mostra como conceder permissão à outra conta para um tópico.

#### Note

Ao conceder à outra Conta da AWS acesso a um recurso em sua conta, você também está concedendo permissões para o recurso em questão aos usuários do IAM com acesso de nível de administrador (acesso coringa). Todos os outros usuários do IAM na outra conta têm automaticamente o acesso negado a seus recursos. Se desejar conceder a usuários específicos do IAM dessa Conta da AWS acesso a seu recurso, a conta ou um usuário do IAM com acesso de nível de administrador deverá delegar permissões para o recurso para esses usuários do IAM. Para obter mais informações sobre a delegação entre contas, consulte [Habilitar acesso entre contas](#) no Guia de uso do IAM.

Se adicionou a seguinte política a um tópico `MeuTópico` na conta, você pode conceder à conta `123456789012` permissão `111122223333` para executar a ação `sns:Publish` no tópico.

```
{
  "Statement": [{
    "Sid": "Allow-publish-to-topic",
    "Effect": "Allow",
    "Principal": {
      "AWS": "111122223333"
    },
    "Action": "sns:Publish",
    "Resource": "arn:aws:sns:us-east-2:123456789012:MyTopic"
  }]
}
```

## Etapa 6: Enviar mensagens para o endpoint HTTP/HTTPS

É possível enviar uma mensagem para inscrições em um tópico publicando no tópico. Para publicar um tópico, use o console do Amazon SNS, o comando [sns-publish](#) da CLI ou a API [Publish](#).

Se seguiu a [Etapa 1](#), o código que você implantou no endpoint deve processar a notificação.

Para publicar um tópico usando o console do Amazon SNS

1. Usando as credenciais da Conta da AWS ou do usuário do IAM com permissão para publicar no tópico, faça login no AWS Management Console e abra o console do Amazon SNS em <https://console.aws.amazon.com/sns/>.
2. No painel de navegação, escolha Topics (Tópicos) e selecione um tópico.
3. Escolha o botão Publicar mensagem.
4. Na caixa Subject (Assunto), insira um assunto (por exemplo, **Testing publish to my endpoint**).
5. Na caixa Message (Mensagem), insira texto (por exemplo, **Hello world!**) e escolha Publish message (Publicar mensagem).

Será exibida a seguinte mensagem: Your message has been successfully published.

## Verificação das assinaturas de mensagens do Amazon SNS

Para verificar a autenticidade de uma mensagem enviada ao seu endpoint HTTP pelo Amazon SNS, você pode verificar a assinatura da mensagem. Há dois casos em que recomendamos verificar a autenticidade da mensagem. Primeiro, quando o Amazon SNS envia uma mensagem ao seu endpoint HTTP informando que você se inscreveu em um tópico. Segundo, quando o Amazon SNS envia uma mensagem de confirmação ao seu endpoint HTTP após a execução das ações `Subscribe` ou `Unsubscribe` da API.

Você deve fazer o seguinte ao verificar mensagens enviadas pelo Amazon SNS:

- Sempre use HTTPS ao obter o certificado do Amazon SNS.
- Valide a autenticidade do certificado.
- Verifique se o certificado foi recebido do Amazon SNS.
- Quando possível, use um dos AWS SDKs compatíveis com o Amazon SNS para validar e verificar as mensagens.

- Confirme se as mensagens do Amazon SNS foram recebidas do `TopicArn` desejado.

O Amazon SNS é compatível com duas versões de assinatura de mensagens:

- `SignatureVersion1`: o Amazon SNS cria a assinatura com base no hash SHA1 da mensagem.
- `SignatureVersion2`: o Amazon SNS cria a assinatura com base no hash SHA256 da mensagem.

Como configurar a versão da assinatura da mensagem nos tópicos do Amazon SNS

Por padrão, os tópicos do Amazon SNS usam `SignatureVersion1`. Para selecionar o algoritmo de hash no tópico do Amazon SNS, `SignatureVersion1` (SHA1) ou `SignatureVersion2` (SHA256), você pode usar a ação `SetTopicAttributes` da API.

O exemplo de código a seguir mostra como definir o atributo do tópico `SignatureVersion` usando a AWS CLI:

```
aws sns set-topic-attributes \  
  --topic-arn arn:aws:sns:us-east-2:123456789012:MyTopic \  
  --attribute-name SignatureVersion \  
  --attribute-value 2
```

Para verificar a assinatura de uma mensagem do Amazon SNS ao usar solicitações baseadas em consulta HTTP

1. Extraia os pares nome/valor do documento JSON no corpo da solicitação HTTP POST enviada pelo Amazon SNS para o endpoint. Você estará usando os valores de alguns dos pares de nome/valor para criar a string para assinar. Ao verificar a assinatura de uma mensagem do Amazon SNS, é fundamental que você converta os caracteres de controle de escape para suas representações de caracteres originais nos valores `Message` e `Subject`. Esses valores devem estar em seus formatos originais quando você os usa como parte da string para assinar. Para obter informações sobre como analisar o documento JSON, consulte [Etapa 1: Verificar se o endpoint está pronto para processar mensagens do Amazon SNS](#).

A `SignatureVersion` informa a versão da assinatura usada pelo Amazon SNS para gerar a assinatura da mensagem. A partir da versão da assinatura, você pode determinar os requisitos para gerar a assinatura. Para notificações, no momento o Amazon SNS é compatível com

a versão de assinatura 1 e 2. Esta seção apresenta as etapas para a verificação de uma assinatura usando essas versões de assinatura.

2. Obtenha o certificado X509 usado pelo Amazon SNS para assinar a mensagem. O valor `SigningCertURL` aponta para a localização do certificado X509 usado para criar a assinatura digital da mensagem. Recupere o certificado nesse local.
3. Extraia a chave pública do certificado. A chave pública do certificado especificado por `SigningCertURL` é usada para verificar a autenticidade e a integridade da mensagem.
4. Determinar o tipo de mensagem. O formato de string a ser assinada depende do tipo de mensagem, que é especificado pelo valor `Type`.
5. Crie a string para assinar. A string a ser assinada é uma lista delimitada por caracteres de nova linha composta por pares de nome-valor específicos da mensagem. Cada par de nome/valor é representado pelo nome seguido por um caractere de nova linha, seguido pelo valor e terminando com um caractere de nova linha. Os pares de nome/valor devem ser listados em ordem de classificação por bytes.

Dependendo do tipo de mensagem, a string para assinar deve ter os seguintes pares de nome/valor.

## Notificação

As mensagens de notificação devem conter os seguintes pares de nome/valor:

```
Message
MessageId
Subject (if included in the message)
Timestamp
TopicArn
Type
```

O exemplo a seguir é uma string para assinar para uma `Notification`.

```
Message
My Test Message
MessageId
4d4dc071-ddbf-465d-bba8-08f81c89da64
Subject
My subject
Timestamp
2019-01-31T04:37:04.321Z
```

```
TopicArn
arn:aws:sns:us-east-2:123456789012:s4-MySNSTopic-1G1WEFC0XTC0P
Type
Notification
```

## SubscriptionConfirmation e UnsubscribeConfirmation

Mensagens de SubscriptionConfirmation e UnsubscribeConfirmation devem conter os seguintes pares de nome/valor:

```
Message
MessageId
SubscribeURL
Timestamp
Token
TopicArn
Type
```

O exemplo a seguir é uma string para assinar para uma SubscriptionConfirmation.

```
Message
My Test Message
MessageId
3d891288-136d-417f-bc05-901c108273ee
SubscribeURL
https://sns.us-east-2.amazonaws.com/?
Action=ConfirmSubscription&TopicArn=arn:aws:sns:us-east-2:123456789012:s4-
MySNSTopic-1G1WEFC0XTC0P&Token=233...
Timestamp
2019-01-31T19:25:13.719Z
Token
233...
TopicArn
arn:aws:sns:us-east-2:123456789012:s4-MySNSTopic-1G1WEFC0XTC0P
Type
SubscriptionConfirmation
```

6. Decodifique o valor Signature do formato Base64. A mensagem oferece a assinatura no valor Signature, que é codificado como Base64. Antes de comparar o valor da assinatura com a assinatura que você calculou, certifique-se de decodificar o valor Signature do Base64 para comparar os valores usando o mesmo formato.

7. Gere o valor de hash derivado da mensagem do Amazon SNS. Envie a mensagem do Amazon SNS, no formato canônico, ao mesmo algoritmo de hash usado para gerar a assinatura.
  - a. Se a `SignatureVersion` for 1, use SHA1 como o algoritmo de hash.
  - b. Se o `SignatureVersion` for 2, use SHA256 como o algoritmo de hash.
8. Gere o valor de hash declarado da mensagem do Amazon SNS. O valor de hash declarado é o resultado de usar o valor de chave pública (da etapa 3) para decodificar a assinatura entregue com a mensagem do Amazon SNS.
9. Verifique a autenticidade e a integridade da mensagem do Amazon SNS. Compare o valor de hash derivado (da etapa 7) com o valor de hash declarado (da etapa 8). Se os valores forem idênticos, o destinatário poderá ter certeza de que a mensagem não foi modificada durante a transferência e que deve ter origem no Amazon SNS. Se os valores não forem idênticos, ela não deve ser confiável para o destinatário.

## Analisar formatos de mensagens

O Amazon SNS usa os formatos a seguir.

### Tópicos

- [Cabeçalhos HTTP/HTTPS](#)
- [Formato JSON de confirmação de assinatura HTTP/HTTPS](#)
- [Formato JSON de notificação HTTP/HTTPS](#)
- [Formato JSON de confirmação de cancelamento de assinatura HTTP/HTTPS](#)
- [Formato JSON da política de entrega `SetSubscriptionAttributes`](#)
- [Formato JSON da política de entrega `SetTopicAttributes`](#)

## Cabeçalhos HTTP/HTTPS

Quando o Amazon SNS envia uma confirmação de assinatura, uma notificação ou uma mensagem de confirmação de cancelamento da assinatura para endpoints HTTP/HTTPS, ele envia uma mensagem POST com uma série de valores de cabeçalho específicos do Amazon SNS. É possível usar valores de cabeçalho para tarefas como identificar o tipo de mensagem sem precisar analisar o corpo da mensagem JSON para ler o valor `Type`. Por padrão, o Amazon SNS envia todas as notificações para endpoints HTTP/S com `Content-Type` definido como `text/plain`;

charset=UTF-8. Para escolher um Content-Type diferente de texto/simples (padrão), consulte `headerContentType` em [Como criar uma política de entrega HTTP/S](#).

### **x-amz-sns-message-type**

O tipo de mensagem. Os valores possíveis são `SubscriptionConfirmation`, `Notification` e `UnsubscribeConfirmation`.

### **x-amz-sns-message-id**

Um Identificador universalmente exclusivo (UUID), exclusivo para cada mensagem publicada. Para uma notificação que o Amazon SNS reenvia durante uma nova tentativa, o ID da mensagem original é usado.

### **x-amz-sns-topic-arn**

O Nome de recurso da Amazon (ARN) para o tópico em que essa mensagem foi publicada.

### **x-amz-sns-subscription-arn**

O ARN para a inscrição desse endpoint.

O cabeçalho HTTP POST a seguir é um exemplo de um cabeçalho para uma mensagem de `Notification` para um endpoint HTTP.

```
POST / HTTP/1.1
x-amz-sns-message-type: Notification
x-amz-sns-message-id: 165545c9-2a5c-472c-8df2-7ff2be2b3b1b
x-amz-sns-topic-arn: arn:aws:sns:us-west-2:123456789012:MyTopic
x-amz-sns-subscription-arn: arn:aws:sns:us-west-2:123456789012:MyTopic:2bcfbf39-05c3-41de-beaa-fcfcc21c8f55
Content-Length: 1336
Content-Type: text/plain; charset=UTF-8
Host: myhost.example.com
Connection: Keep-Alive
User-Agent: Amazon Simple Notification Service Agent
```

## Formato JSON de confirmação de assinatura HTTP/HTTPS

Depois de você inscrever um endpoint HTTP/HTTPS, o Amazon SNS enviará uma mensagem de confirmação de inscrição para o endpoint HTTP/HTTPS. Essa mensagem contém um valor `SubscribeURL` que você deve consultar para confirmar a assinatura (como alternativa, é possível usar o valor `Token` com o [ConfirmSubscription](#)).

**Note**

O Amazon SNS não enviará notificações para esse endpoint enquanto a inscrição não for confirmada.

A mensagem de confirmação da inscrição é uma mensagem POST com um corpo de mensagem que contém um documento JSON com os pares de nome/valor a seguir.

**Type**

O tipo de mensagem. Para uma confirmação da inscrição, o tipo é `SubscriptionConfirmation`.

**MessageId**

Um Identificador universalmente exclusivo (UUID), exclusivo para cada mensagem publicada. Para uma mensagem que o Amazon SNS reenvia durante uma nova tentativa, o ID da mensagem original é usado.

**Token**

Um valor que pode ser usado com a ação [ConfirmSubscription](#) para confirmar a assinatura. Como alternativa, você pode simplesmente acessar o `SubscribeURL`.

**TopicArn**

O Nome de recurso da Amazon (ARN) para o tópico em que este endpoint está inscrito.

**Message**

Uma string que descreve a mensagem. Para confirmação da inscrição, esta string é semelhante a:

```
You have chosen to subscribe to the topic arn:aws:sns:us-east-2:123456789012:MyTopic.\n\nTo confirm the subscription, visit the SubscribeURL included in this message.
```

**SubscribeURL**

O URL que você deve acessar para confirmar a inscrição. Como alternativa, é possível usar o `Token` com a ação [ConfirmSubscription](#) para confirmar a assinatura.

## Timestamp

A hora (GMT) quando a confirmação da inscrição foi enviada.

## SignatureVersion

Versão da assinatura do Amazon SNS usada.

- Se `SignatureVersion` for 1, `Signature` será uma assinatura SHA1withRSA codificada em Base64 dos valores `Message`, `MessageId`, `Type`, `Timestamp` e `TopicArn`.
- Se `SignatureVersion` for 2, `Signature` será uma assinatura SHA256withRSA codificada em Base64 dos valores `Message`, `MessageId`, `Type`, `Timestamp` e `TopicArn`.

## Signature

Assinatura SHA1withRSA ou SHA256withRSA codificada em Base64 dos valores `Message`, `MessageId`, `Type`, `Timestamp` e `TopicArn`.

## SigningCertURL

O URL do certificado que foi usado para assinar a mensagem.

A mensagem HTTP POST a seguir é um exemplo de uma mensagem de `SubscriptionConfirmation` para um endpoint HTTP.

```
POST / HTTP/1.1
x-amz-sns-message-type: SubscriptionConfirmation
x-amz-sns-message-id: 165545c9-2a5c-472c-8df2-7ff2be2b3b1b
x-amz-sns-topic-arn: arn:aws:sns:us-west-2:123456789012:MyTopic
Content-Length: 1336
Content-Type: text/plain; charset=UTF-8
Host: myhost.example.com
Connection: Keep-Alive
User-Agent: Amazon Simple Notification Service Agent

{
  "Type" : "SubscriptionConfirmation",
  "MessageId" : "165545c9-2a5c-472c-8df2-7ff2be2b3b1b",
  "Token" : "2336412f37...",
  "TopicArn" : "arn:aws:sns:us-west-2:123456789012:MyTopic",
  "Message" : "You have chosen to subscribe to the topic arn:aws:sns:us-
west-2:123456789012:MyTopic.\nTo confirm the subscription, visit the SubscribeURL
included in this message.",
```

```
"SubscribeURL" : "https://sns.us-west-2.amazonaws.com/?
Action=ConfirmSubscription&TopicArn=arn:aws:sns:us-
west-2:123456789012:MyTopic&Token=2336412f37...",
"Timestamp" : "2012-04-26T20:45:04.751Z",
"SignatureVersion" : "1",
"Signature" : "EXAMPLEpH
+DcEwjAPg809mY8dReBSwksfg2S7WKQcikcNKWLQjwu6A4VbeS0QHVCkhRS7fUQvi2egU3N858fiTDN6bkk0xYDVrY0Ad8L
"SigningCertURL" : "https://sns.us-west-2.amazonaws.com/SimpleNotificationService-
f3ecfb7224c7233fe7bb5f59f96de52f.pem"
}
```

## Formato JSON de notificação HTTP/HTTPS

Quando o Amazon SNS envia uma notificação para um endpoint HTTP ou HTTPS inscrito, a mensagem POST enviada para o endpoint tem um corpo da mensagem que contém um documento JSON com os seguintes pares de nome-valor.

### Type

O tipo de mensagem. Para uma notificação, o tipo é `Notification`.

### MessageId

Um Identificador universalmente exclusivo (UUID), exclusivo para cada mensagem publicada. Para uma notificação que o Amazon SNS reenvia durante uma nova tentativa, o ID da mensagem original é usado.

### TopicArn

O Nome de recurso da Amazon (ARN) para o tópico em que essa mensagem foi publicada.

### Subject

O parâmetro `Subject` especificado quando a notificação foi publicada no tópico.

#### Note

Esse parâmetro é opcional. Se nenhum `Subject` foi especificado, esse par de nome/valor não é exibido nesse documento JSON.

### Message

O valor `Message` especificado quando a notificação foi publicada no tópico.

## Timestamp

A hora (GMT) quando a notificação foi publicada.

## SignatureVersion

Versão da assinatura do Amazon SNS usada.

- Se `SignatureVersion` for 1, `Signature` será uma assinatura SHA1withRSA codificada em Base64 dos valores `Message`, `MessageId`, `Subject` (se presente), `Type`, `Timestamp` e `TopicArn`.
- Se `SignatureVersion` for 2, `Signature` será uma assinatura SHA256withRSA codificada em Base64 dos valores `Message`, `MessageId`, `Subject` (se presente), `Type`, `Timestamp` e `TopicArn`.

## Signature

Assinatura SHA1withRSA ou SHA256withRSA codificada em Base64 dos valores `Message`, `MessageId`, `Subject` (se presente), `Type`, `Timestamp` e `TopicArn`.

## SigningCertURL

O URL do certificado que foi usado para assinar a mensagem.

## UnsubscribeURL

Um URL que você pode usar para cancelar a inscrição do endpoint com base neste tópico. Se você acessar este URL, o Amazon SNS cancela a inscrição do endpoint e interrompe o envio de notificações para esse endpoint.

A mensagem HTTP POST a seguir é um exemplo de uma mensagem de `Notification` para um endpoint HTTP.

```
POST / HTTP/1.1
x-amz-sns-message-type: Notification
x-amz-sns-message-id: 22b80b92-fdea-4c2c-8f9d-bdfb0c7bf324
x-amz-sns-topic-arn: arn:aws:sns:us-west-2:123456789012:MyTopic
x-amz-sns-subscription-arn: arn:aws:sns:us-west-2:123456789012:MyTopic:c9135db0-26c4-47ec-8998-413945fb5a96
Content-Length: 773
Content-Type: text/plain; charset=UTF-8
Host: myhost.example.com
```

```
Connection: Keep-Alive
User-Agent: Amazon Simple Notification Service Agent

{
  "Type" : "Notification",
  "MessageId" : "22b80b92-fdea-4c2c-8f9d-bdfb0c7bf324",
  "TopicArn" : "arn:aws:sns:us-west-2:123456789012:MyTopic",
  "Subject" : "My First Message",
  "Message" : "Hello world!",
  "Timestamp" : "2012-05-02T00:54:06.655Z",
  "SignatureVersion" : "1",
  "Signature" : "EXAMPLEw6JRN...",
  "SigningCertURL" : "https://sns.us-west-2.amazonaws.com/SimpleNotificationService-
f3ecfb7224c7233fe7bb5f59f96de52f.pem",
  "UnsubscribeURL" : "https://sns.us-west-2.amazonaws.com/?
Action=Unsubscribe&SubscriptionArn=arn:aws:sns:us-
west-2:123456789012:MyTopic:c9135db0-26c4-47ec-8998-413945fb5a96"
}
```

## Formato JSON de confirmação de cancelamento de assinatura HTTP/HTTPS

Depois que a inscrição de um endpoint HTTP/HTTPS é cancelada em um tópico, o Amazon SNS envia uma mensagem de confirmação de cancelamento de inscrição para o endpoint.

A mensagem de confirmação do cancelamento da inscrição é uma mensagem POST com um corpo de mensagem que contém um documento JSON com os pares de nome/valor a seguir.

### Type

O tipo de mensagem. Para uma confirmação de cancelamento de inscrição, o tipo é `UnsubscribeConfirmation`.

### MessageId

Um Identificador universalmente exclusivo (UUID), exclusivo para cada mensagem publicada. Para uma mensagem que o Amazon SNS reenvia durante uma nova tentativa, o ID da mensagem original é usado.

### Token

Um valor que pode ser usado com a ação [ConfirmSubscription](#) para confirmar novamente a assinatura. Como alternativa, você pode simplesmente acessar o `SubscribeURL`.

## TopicArn

O Nome de recurso da Amazon (ARN) para o tópico do qual esse endpoint teve sua inscrição cancelada.

## Message

Uma string que descreve a mensagem. Para a confirmação de cancelamento de inscrição, esta string é semelhante a:

```
You have chosen to deactivate subscription arn:aws:sns:us-east-2:123456789012:MyTopic:2bcfbf39-05c3-41de-beaa-fcfcc21c8f55.\n\nTo cancel this operation and restore the subscription, visit the SubscribeURL included in this message.
```

## SubscribeURL

O URL que você deve acessar para confirmar novamente a inscrição. Como alternativa, é possível usar o Token com a ação [ConfirmSubscription](#) para confirmar novamente a assinatura.

## Timestamp

A hora (GMT) quando a confirmação de cancelamento da inscrição foi enviada.

## SignatureVersion

Versão da assinatura do Amazon SNS usada.

- Se `SignatureVersion` for 1, `Signature` será uma assinatura SHA1withRSA codificada em Base64 dos valores `Message`, `MessageId`, `Type`, `Timestamp` e `TopicArn`.
- Se `SignatureVersion` for 2, `Signature` será uma assinatura SHA256withRSA codificada em Base64 dos valores `Message`, `MessageId`, `Type`, `Timestamp` e `TopicArn`.

## Signature

Assinatura SHA1withRSA ou SHA256withRSA codificada em Base64 dos valores `Message`, `MessageId`, `Type`, `Timestamp` e `TopicArn`.

## SigningCertURL

O URL do certificado que foi usado para assinar a mensagem.

A mensagem HTTP POST a seguir é um exemplo de uma mensagem de `UnsubscribeConfirmation` para um endpoint HTTP.

```
POST / HTTP/1.1
x-amz-sns-message-type: UnsubscribeConfirmation
x-amz-sns-message-id: 47138184-6831-46b8-8f7c-afc488602d7d
x-amz-sns-topic-arn: arn:aws:sns:us-west-2:123456789012:MyTopic
x-amz-sns-subscription-arn: arn:aws:sns:us-
west-2:123456789012:MyTopic:2bcfbf39-05c3-41de-beaa-fcfcc21c8f55
Content-Length: 1399
Content-Type: text/plain; charset=UTF-8
Host: myhost.example.com
Connection: Keep-Alive
User-Agent: Amazon Simple Notification Service Agent

{
  "Type" : "UnsubscribeConfirmation",
  "MessageId" : "47138184-6831-46b8-8f7c-afc488602d7d",
  "Token" : "2336412f37...",
  "TopicArn" : "arn:aws:sns:us-west-2:123456789012:MyTopic",
  "Message" : "You have chosen to deactivate subscription arn:aws:sns:us-
west-2:123456789012:MyTopic:2bcfbf39-05c3-41de-beaa-fcfcc21c8f55.\nTo cancel this
operation and restore the subscription, visit the SubscribeURL included in this
message.",
  "SubscribeURL" : "https://sns.us-west-2.amazonaws.com/?
Action=ConfirmSubscription&TopicArn=arn:aws:sns:us-
west-2:123456789012:MyTopic&Token=2336412f37fb6...",
  "Timestamp" : "2012-04-26T20:06:41.581Z",
  "SignatureVersion" : "1",
  "Signature" : "EXAMPLEHXgJm...",
  "SigningCertURL" : "https://sns.us-west-2.amazonaws.com/SimpleNotificationService-
f3ecfb7224c7233fe7bb5f59f96de52f.pem"
}
```

## Formato JSON da política de entrega SetSubscriptionAttributes

Se você envia uma solicitação para a ação `SetSubscriptionAttributes` e define o parâmetro `AttributeName` para um valor de `DeliveryPolicy`, o valor do parâmetro `AttributeValue` deve ser um objeto JSON válido. Por exemplo, o exemplo a seguir define a política de entrega para um total de cinco novas tentativas.

```
http://sns.us-east-2.amazonaws.com/
?Action=SetSubscriptionAttributes
&SubscriptionArn=arn%3Aaws%3Asns%3Aus-east-2%3A123456789012%3AMy-Topic
%3A80289ba6-0fd4-4079-afb4-ce8c8260f0ca
```

```
&AttributeName=DeliveryPolicy
&AttributeValue={"healthyRetryPolicy":{"numRetries":5}}
...
```

Use o formato JSON a seguir para o valor do parâmetro `AttributeValue`.

```
{
  "healthyRetryPolicy" : {
    "minDelayTarget" : int,
    "maxDelayTarget" : int,
    "numRetries" : int,
    "numMaxDelayRetries" : int,
    "backoffFunction" : "linear|arithmetic|geometric|exponential"
  },
  "throttlePolicy" : {
    "maxReceivesPerSecond" : int
  },
  "requestPolicy" : {
    "headerContentType" : "text/plain | application/json | application/xml"
  }
}
```

Para obter mais informações sobre a ação `SetSubscriptionAttribute`, consulte [SetSubscriptionAttributes](#) na Referência de API do Amazon Simple Notification Service. Para obter mais informações sobre os cabeçalhos do tipo de conteúdo HTTP compatíveis, consulte [Como criar uma política de entrega HTTP/S](#).

## Formato JSON da política de entrega `SetTopicAttributes`

Se você envia uma solicitação para a ação `SetTopicAttributes` e define o parâmetro `AttributeName` para um valor de `DeliveryPolicy`, o valor do parâmetro `AttributeValue` deve ser um objeto JSON válido. Por exemplo, o exemplo a seguir define a política de entrega para um total de cinco novas tentativas.

```
http://sns.us-east-2.amazonaws.com/
?Action=SetTopicAttributes
&TopicArn=arn%3Aaws%3Asns%3Aus-east-2%3A123456789012%3AMy-Topic
&AttributeName=DeliveryPolicy
&AttributeValue={"http":{"defaultHealthyRetryPolicy":{"numRetries":5}}}
...
```

Use o formato JSON a seguir para o valor do parâmetro `AttributeValue`.

```
{
  "http" : {
    "defaultHealthyRetryPolicy" : {
      "minDelayTarget": int,
      "maxDelayTarget": int,
      "numRetries": int,
      "numMaxDelayRetries": int,
      "backoffFunction": "linear|arithmetic|geometric|exponential"
    },
    "disableSubscriptionOverrides" : Boolean,
    "defaultThrottlePolicy" : {
      "maxReceivesPerSecond" : int
    },
    "defaultRequestPolicy" : {
      "headerContentType" : "text/plain | application/json | application/xml"
    }
  }
}
```

Para obter mais informações sobre a ação `SetTopicAttribute`, consulte

[SetSubscriptionAttributes](#) na Referência da API do Amazon Simple Notification Service. Para obter mais informações sobre os cabeçalhos do tipo de conteúdo HTTP compatíveis, consulte [Como criar uma política de entrega HTTP/S](#).

## Fanout para AWS Event Fork Pipelines

Para arquivamento e análise de eventos, o Amazon SNS agora recomenda usar sua integração nativa com o Amazon Data Firehose. Você pode inscrever streams de entrega do Firehose em tópicos do SNS, o que permite enviar notificações para endpoints de arquivamento e análise, como buckets do Amazon Simple Storage Service (Amazon S3), tabelas do Amazon Redshift, Amazon Service (Service) e muito mais. OpenSearch Usar o Amazon SNS com os streams de entrega do Firehose é uma solução totalmente gerenciada e sem código que não exige o uso de funções. AWS Lambda Para ter mais informações, consulte [Fanout para streams de entrega do Firehose](#).

Você pode usar o Amazon SNS para desenvolver aplicações orientadas por eventos que usam serviços de assinante para executar o trabalho automaticamente em resposta a eventos acionados por serviços do editor. Esse padrão de arquitetura pode tornar os serviços mais reutilizáveis, interoperáveis e dimensionáveis. No entanto, pode ser trabalhoso usar o processamento de eventos em pipelines que abordam os requisitos de manipulação de eventos comuns, como armazenamento de eventos, backup, pesquisa, análise e reprodução.

Para acelerar o desenvolvimento de suas aplicações orientadas por eventos, você pode assinar pipelines de tratamento de eventos, por meio do AWS Event Fork Pipelines, em tópicos do Amazon SNS. O AWS Event Fork Pipelines é um conjunto de [aplicações aninhadas](#) de código aberto, com base no [AWS Serverless Application Model](#) (AWS SAM), que você pode implantar diretamente do pacote [AWS Event Fork Pipelines](#) (escolha Show apps that create custom IAM roles or resource policies (Mostrar aplicações que criam funções personalizadas do IAM ou políticas de recurso) em sua conta da AWS.

Para um caso de uso do AWS Event Fork Pipelines, consulte [Implantar e testar a aplicação de exemplo do AWS Event Fork Pipelines](#).

### Tópicos

- [Como o AWS Event Fork Pipelines funciona](#)
- [Implantar o AWS Event Fork Pipelines](#)
- [Implantar e testar a aplicação de exemplo do AWS Event Fork Pipelines](#)
- [Inscrever um AWS Event Fork Pipelines em um tópico do Amazon SNS](#)

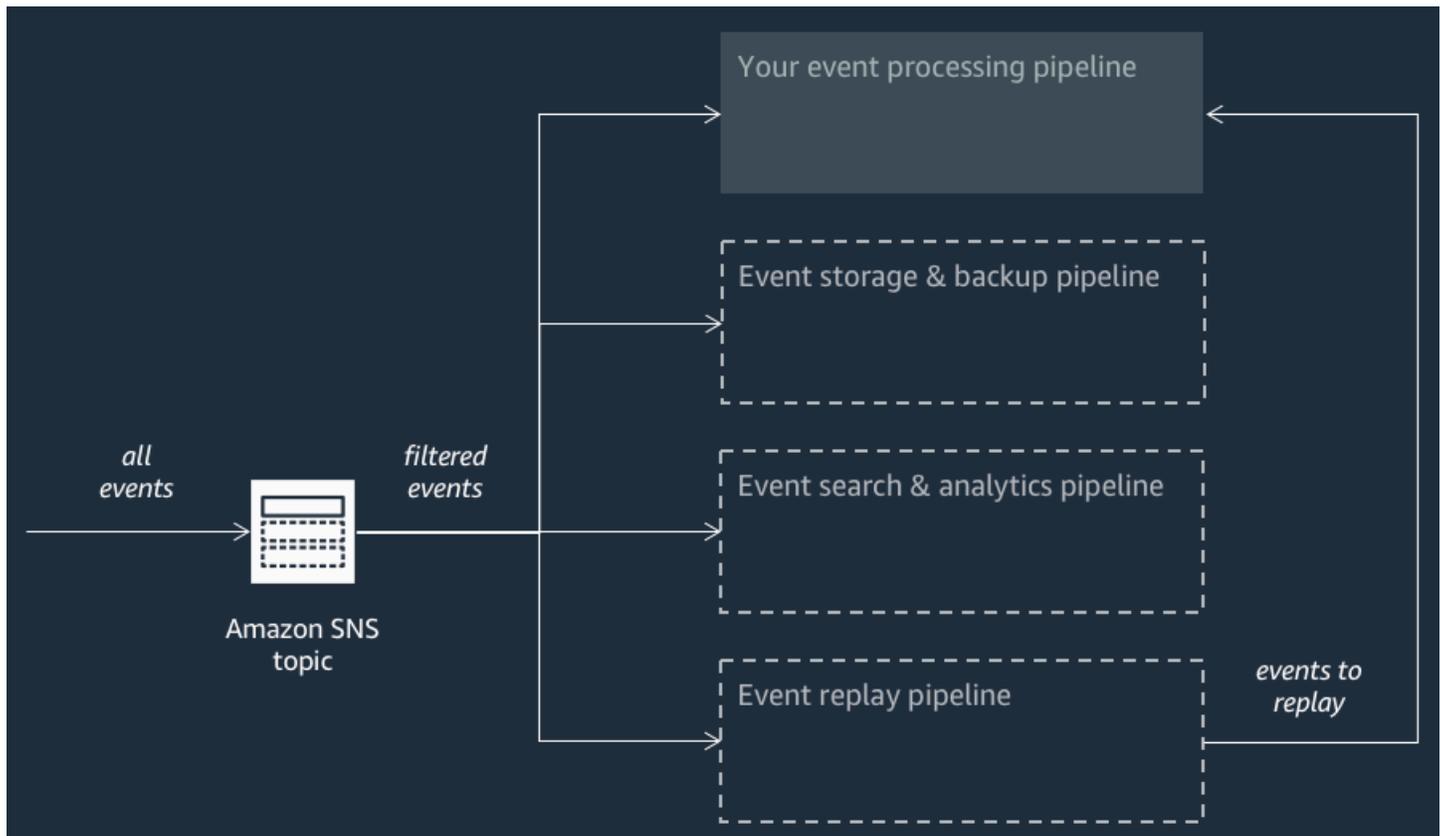
## Como o AWS Event Fork Pipelines funciona

O AWS Event Fork Pipelines é um padrão de projeto sem servidor. No entanto, ele também é um pacote de aplicações aninhadas sem servidor, baseado no AWS SAM (que pode ser implantado diretamente do AWS Serverless Application Repository (AWS SAR) em sua Conta da AWS para aprimorar suas plataformas orientadas por eventos). Você pode implantar essas aplicações aninhadas individualmente, conforme sua arquitetura exigir.

### Tópicos

- [O pipeline de armazenamento e backup de eventos](#)
- [O pipeline de pesquisa e análise de eventos](#)
- [O pipeline de reprodução de eventos](#)

O diagrama a seguir mostra uma aplicação do AWS Event Fork Pipelines complementada por três aplicações aninhadas. É possível implantar qualquer um dos pipelines do pacote do AWS Event Fork Pipelines no AWS SAR de maneira independente, conforme sua arquitetura exigir.



Cada pipeline é inscrito no mesmo tópico do Amazon SNS, o que permite que ele processe eventos em paralelo conforme esses eventos são publicados no tópico. Cada pipeline é independente e pode definir sua própria [política de filtro de assinatura](#). Isso permite que um pipeline processe apenas um subconjunto dos eventos em que está interessado (em vez de todos os eventos publicados no tópico).

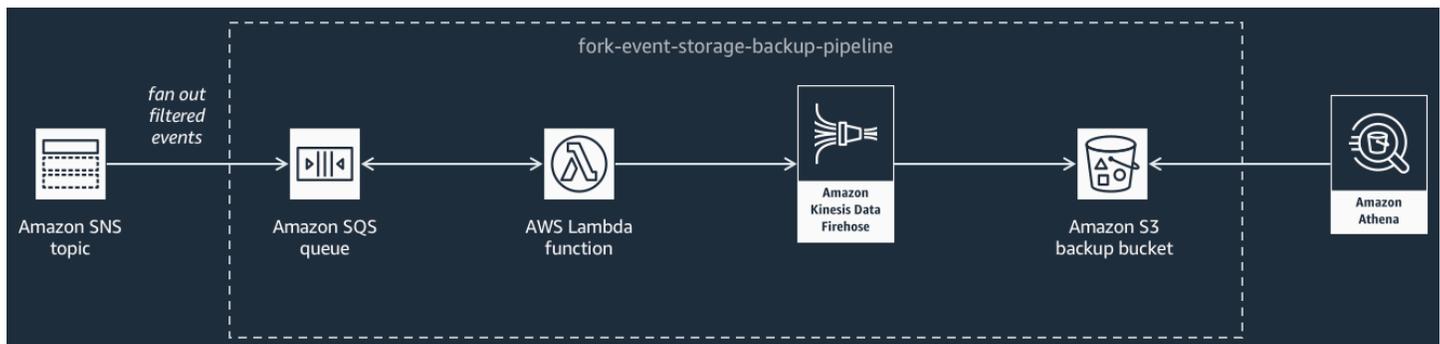
#### **Note**

Como os três AWS Event Fork Pipelines são colocados juntos com os pipelines de processamento de eventos usuais (possivelmente já inscritos em seu tópico do Amazon SNS), não é necessário alterar nenhuma parte do editor de mensagens atual para aproveitar o AWS Event Fork Pipelines nas workloads existentes.

## O pipeline de armazenamento e backup de eventos

O diagrama a seguir mostra o [pipeline de armazenamento e backup de eventos](#). Você pode inscrever esse pipeline em seu tópico do Amazon SNS para fazer backup automaticamente dos eventos que passam pelo sistema.

Esse pipeline é composto por uma fila do Amazon SQS que armazena em buffer os eventos entregues pelo tópico do Amazon SNS, AWS Lambda uma função que pesquisa automaticamente esses eventos na fila e os envia para um stream do Amazon Data Firehose e um bucket do Amazon S3 que faz backup duradouro dos eventos carregados pelo stream.

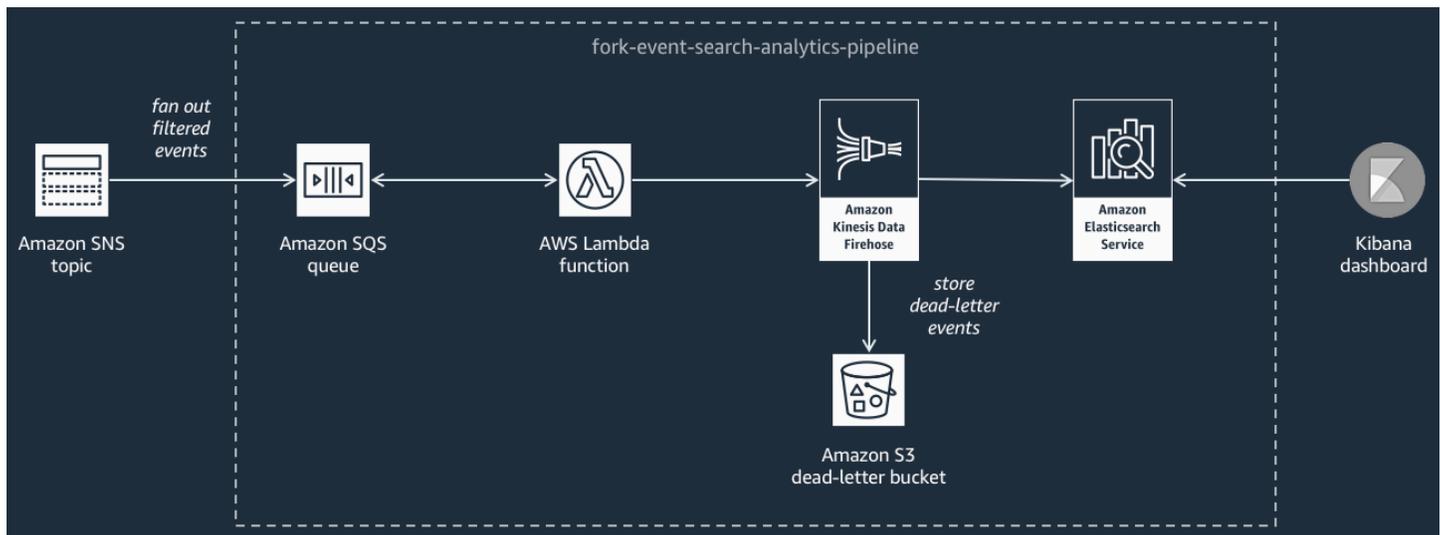


Para ajustar o comportamento de seu fluxo do Firehose, configure-o para armazenar em buffer, transformar e compactar seus eventos antes de carregá-los no bucket. À medida que os eventos forem carregados, use o Amazon Athena para consultar o bucket usando consultas SQL padrão. Você também pode configurar o pipeline para reutilizar um bucket do Simple Storage Service (Amazon S3) existente ou criar um novo.

## O pipeline de pesquisa e análise de eventos

O diagrama a seguir mostra o [pipeline de pesquisa e análise de eventos](#). Você pode inscrever esse pipeline em seu tópico do Amazon SNS para indexar os eventos que passam pelo sistema em um domínio de pesquisa e analisá-los.

Esse pipeline é composto por uma fila do Amazon SQS que armazena em buffer os eventos entregues pelo tópico do Amazon SNS, AWS Lambda uma função que pesquisa eventos da fila e os envia para um stream do Amazon Data Firehose, um domínio do Amazon OpenSearch Service que indexa os eventos carregados pelo stream do Firehose e um bucket do Amazon S3 que armazena os eventos sem saída que não podem ser indexado no domínio de pesquisa.



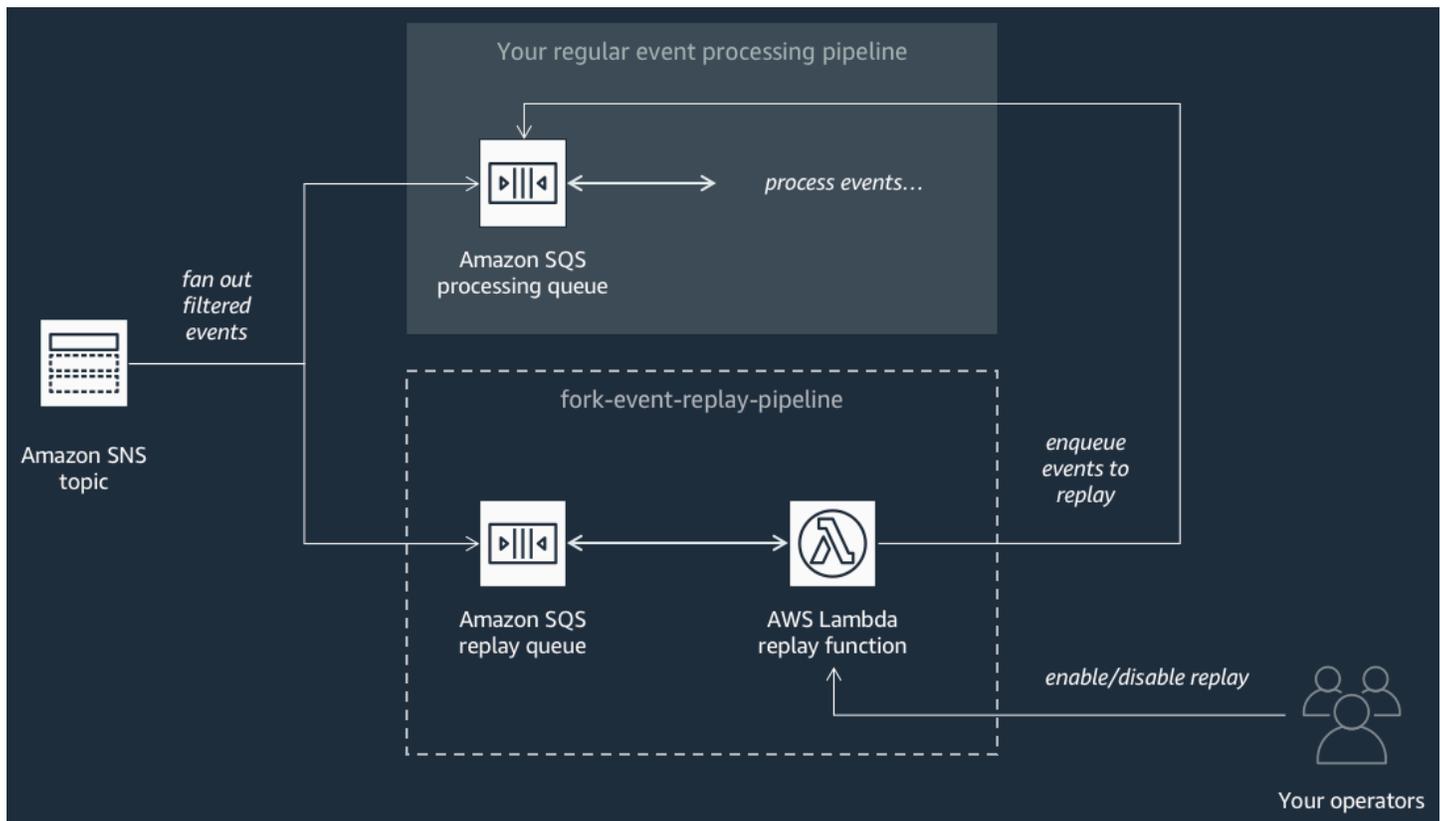
Para ajustar o fluxo do Firehose em termos de armazenamento de eventos em buffer, transformação e compactação, você pode configurar esse pipeline.

Você também pode configurar se o pipeline deve reutilizar um OpenSearch domínio existente no seu Conta da AWS ou criar um novo para você. À medida que os eventos forem indexados no domínio de pesquisa, use o Kibana para executar a análise de seus eventos e atualizar os painéis visuais em tempo real.

## O pipeline de reprodução de eventos

O diagrama a seguir mostra o [pipeline de reprodução de eventos](#). Para registrar os eventos que foram processados pelo sistema nos últimos 14 dias (por exemplo, quando sua plataforma precisar se recuperar de uma falha), inscreva esse pipeline em seu tópico do Amazon SNS e processe novamente os eventos.

Esse pipeline é composto de uma fila do Amazon SQS que armazena em buffer os eventos entregues pelo tópico do Amazon SNS e uma função do AWS Lambda que sonda os eventos da fila e os redireciona para o pipeline de processamento de eventos usual, que também está inscrito em seu tópico.



### Note

Por padrão, a função de reprodução está desabilitada e não redireciona seus eventos. Se precisar reprocessar os eventos, você deve habilitar a fila de reprodução do Amazon SQS como uma origem de eventos para a função de reprodução do AWS Lambda.

## Implantar o AWS Event Fork Pipelines

O [pacote AWS Event Fork Pipelines](#) (escolha Show apps that create custom IAM roles or resource policies (Mostrar aplicações que criam funções do IAM ou políticas de recurso personalizadas)) está disponível como um grupo de aplicações públicas no AWS Serverless Application Repository, de onde você pode implantá-las e testá-las manualmente usando o [console do AWS Lambda](#). Para obter informações sobre como implantar pipelines usando o console do AWS Lambda, consulte [Inscrever um AWS Event Fork Pipelines em um tópico do Amazon SNS](#).

Em um cenário de produção, recomendamos a incorporação do AWS Event Fork Pipelines no modelo do AWS SAM de sua aplicação geral. O recurso de aplicação aninhada permite fazer isso

ao adicionar o recurso [AWS::Serverless::Application](#) ao seu modelo do AWS SAM, fazendo referência ao AWS e à ApplicationId da aplicação aninhada do SemanticVersion SAR.

Por exemplo, use o pipeline de armazenamento e backup de eventos como uma aplicação aninhada adicionando o seguinte trecho de YAML à seção Resources de seu modelo do AWS SAM.

```
Backup:
  Type: AWS::Serverless::Application
  Properties:
    Location:
      ApplicationId: arn:aws:serverlessrepo:us-east-2:123456789012:applications/fork-
event-storage-backup-pipeline
      SemanticVersion: 1.0.0
    Parameters:
      #The ARN of the Amazon SNS topic whose messages should be backed up to the Amazon
S3 bucket.
      TopicArn: !Ref MySNSTopic
```

Ao especificar valores de parâmetro, use funções intrínsecas do AWS CloudFormation para fazer referência a outros recursos em seu modelo. Por exemplo, no trecho YAML acima, o parâmetro TopicArn faz referência ao recurso MySNSTopic do [AWS::SNS::Topic](#), definido em outro lugar no modelo do AWS SAM. Para obter mais informações, consulte o [Referência à função intrínseca](#) no Guia do usuário do usuário do AWS CloudFormation.

#### Note

A página do console do AWS Lambda para a aplicação do AWS SAR inclui o botão Copy as SAM Resource (Copiar como recurso do SAM), que copia o YAML necessário para aninhar uma aplicação do AWS SAR à área de transferência.

## Implantar e testar a aplicação de exemplo do AWS Event Fork Pipelines

Para acelerar o desenvolvimento de suas aplicações orientadas a eventos, você pode inscrever pipelines de tratamento de eventos, com a tecnologia AWS Event Fork Pipelines, em tópicos do Amazon SNS. AWS Event Fork Pipelines é um conjunto de [aplicativos aninhados](#) de código aberto, com base no [AWS Serverless Application Model](#) (AWS SAM), que você pode implantar diretamente do pacote [AWS Event Fork Pipelines](#) (escolha Show apps that create custom IAM roles or resource policies (Mostrar aplicativos que criam funções personalizadas do IAM ou políticas de recursos) em sua conta da AWS. Para ter mais informações, consulte [Como o AWS Event Fork Pipelines funciona](#).

Esta página mostra como usar o AWS Management Console para implantar e testar a aplicação de exemplo do AWS Event Fork Pipelines

### Important

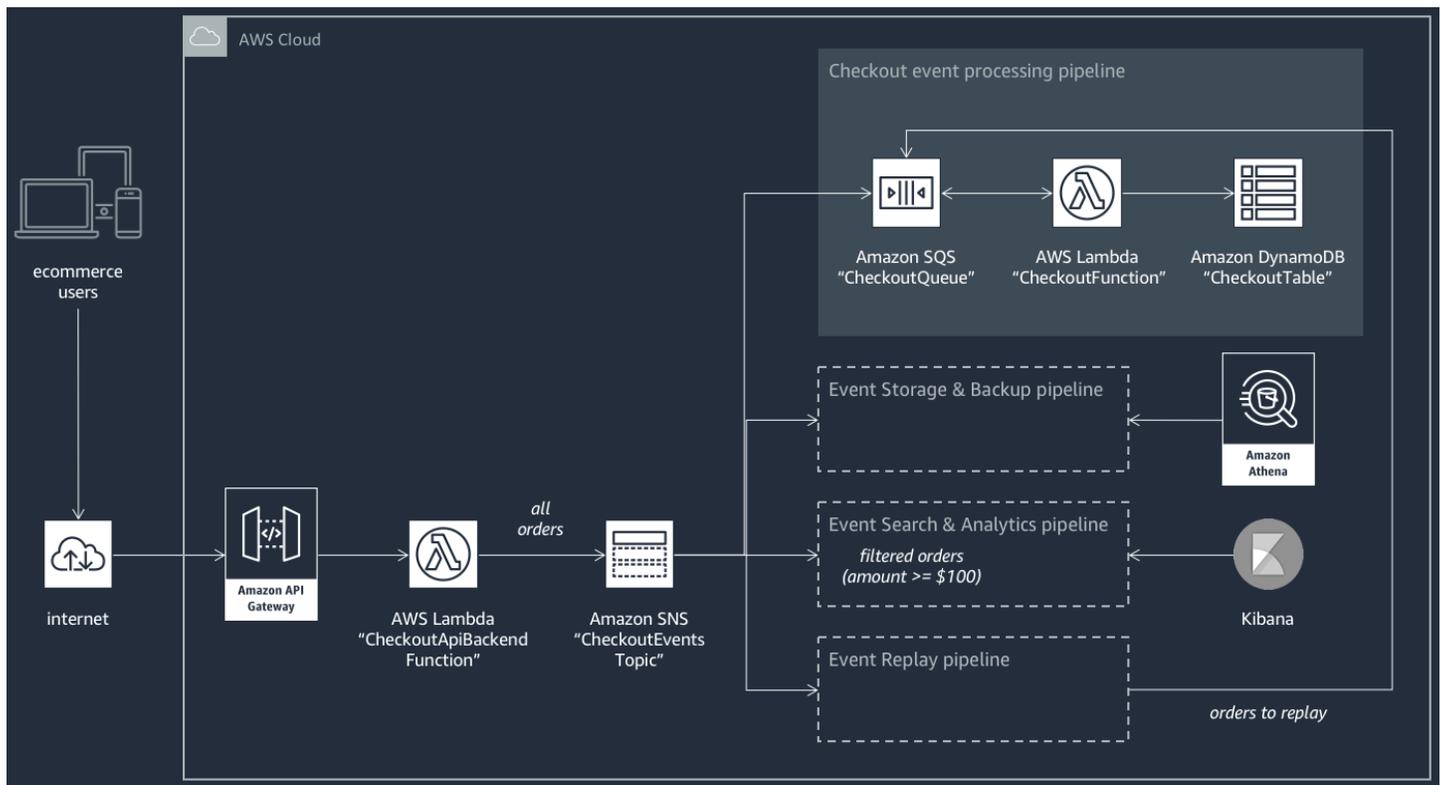
Para evitar a incidência de custos indesejados após concluir a implantação da aplicação de exemplo do AWS Event Fork Pipelines, exclua sua pilha do AWS CloudFormation. Para obter mais informações, consulte [Excluir uma pilha no console do AWS CloudFormation](#) no Manual do usuário do AWS CloudFormation.

## Tópicos

- [Exemplo de caso de uso do AWS Event Fork Pipelines](#)
- [Etapa 1: Como implantar o aplicativo de exemplo](#)
- [Etapa 2: Como executar o aplicativo de exemplo](#)
- [Etapa 3: Como verificar a execução do aplicativo de exemplo e seus pipelines](#)
- [Etapa 4: Como simular um problema e repetir eventos para recuperação](#)

## Exemplo de caso de uso do AWS Event Fork Pipelines

O cenário a seguir descreve um aplicativo de comércio eletrônico sem servidor e orientado por eventos que usa o AWS Event Fork Pipelines. Você pode usar esse [exemplo de aplicativo de comércio eletrônico](#) no AWS Serverless Application Repository e, em seguida, implantá-lo no AWS Lambda console, onde você pode testá-lo e examinar seu código-fonte GitHub. Conta da AWS



Essa aplicação de comércio eletrônico recebe pedidos de compradores por meio de uma API RESTful hospedada pelo API Gateway e respaldada pela função `CheckoutApiBackendFunction` do AWS Lambda. Essa função publica todos os pedidos recebidos em um tópico do Amazon SNS chamado `CheckoutEventsTopic` que, por sua vez, envia os pedidos para quatro pipelines diferentes.

O primeiro pipeline é o pipeline de processamento de checkout usual projetado e implementado pelo proprietário do aplicativo de comércio eletrônico. Esse pipeline tem a fila `CheckoutQueue` do Amazon SQS que armazena em buffer todos os pedidos recebidos, uma função do AWS Lambda chamada `CheckoutFunction` que sonda a fila para processar esses pedidos e a tabela `CheckoutTable` do DynamoDB que salva com segurança todos os pedidos.

### Aplicar o AWS Event Fork Pipelines

Os componentes do aplicativo de comércio eletrônico processam a lógica de negócios central. No entanto, o proprietário do aplicativo de comércio eletrônico também precisa abordar o seguinte:

- **Conformidade:** backups seguros e compactados, criptografados em repouso e limpeza de informações confidenciais
- **Resiliência:** repetição dos pedidos mais recentes em caso de interrupção do processo de atendimento

- Capacidade de pesquisa: execução de análise e geração de métricas nos pedidos realizados

Em vez de implementar essa lógica de processamento de eventos, o proprietário do aplicativo pode inscrever o AWS Event Fork Pipelines no tópico `CheckoutEventsTopic` do Amazon SNS

- O [O pipeline de armazenamento e backup de eventos](#) é configurado com o intuito de transformar dados para remover detalhes de cartões de crédito, armazenar os dados em buffer durante 60 segundos, compactá-los usando GZIP e criptografá-los usando a chave mestra de cliente (CMK) padrão do Amazon S3. Essa chave é gerenciada pela AWS e conta com recursos do AWS Key Management Service (AWS KMS).

Para obter mais informações, consulte [Escolha o Amazon S3 para seu destino](#), [Transformação de dados do Amazon Data Firehose](#) e [Definir configurações no](#) Guia do desenvolvedor do Amazon Data Firehose.

- O [O pipeline de pesquisa e análise de eventos](#) é configurado com uma duração de nova tentativa do índice de 30 segundos, um bucket para armazenar os pedidos que não são indexados no domínio de pesquisa e uma política de filtro para restringir o conjunto de pedidos indexados.

Para obter mais informações, consulte [Escolha o OpenSearch serviço para seu destino](#) no Guia do desenvolvedor do Amazon Data Firehose.

- O [O pipeline de reprodução de eventos](#) é configurado com a parte da fila do Amazon SQS do pipeline de processamento de pedidos usual projetado e implementado pela proprietária da aplicação de comércio eletrônico.

Para obter mais informações, consulte [Queue Name and URL](#) (“Nome e URL da fila”) no Guia do desenvolvedor do Amazon Simple Queue Service.

A seguinte política de filtro JSON é definida na configuração do pipeline de pesquisa e análise de eventos. Ela corresponde apenas aos pedidos de entrada em que a quantidade total for de 100 USD ou superior. Para ter mais informações, consulte [Filtragem de mensagens do Amazon SNS](#).

```
{
  "amount": [ { "numeric": [ ">=", 100 ] } ]
}
```

Ao usar o padrão AWS Event Fork Pipelines, a proprietária da aplicação de comércio eletrônico pode evitar a sobrecarga de desenvolvimento que normalmente acompanha a lógica indiferenciada

de codificação para o tratamento de eventos. Em vez disso, ela pode implantar AWS Event Fork Pipelines diretamente do AWS Serverless Application Repository na Conta da AWS.

## Etapa 1: Como implantar o aplicativo de exemplo

1. Faça login no [AWS Lambdaconsole](#).
2. No painel de navegação, escolha Functions (Funções) e selecione Create function (Criar função).
3. Na página Create function (Criar função), faça o seguinte:
  - a. Escolha Browse serverless app repository (Procurar no repositório de aplicações sem servidor), Public applications (Aplicações públicas), Show apps that create custom IAM roles or resource policies (Mostrar aplicações que criam funções do IAM ou políticas de recursos).
  - b. Procure `fork-example-ecommerce-checkout-api` e escolha o aplicativo.
4. Na página `fork-example-ecommerce-checkout-api`, faça o seguinte:
  - a. Na seção Application settings (Configurações do aplicativo), insira um Application name (Nome de aplicativo) (por exemplo, `fork-example-ecommerce-my-app`).

### Note

- Para encontrar facilmente seus recursos posteriormente, mantenha o prefixo `fork-example-ecommerce`.
- Para cada implantação, o nome do aplicativo deve ser exclusivo. Se você reutilizar um nome de aplicativo, a implantação atualizará apenas a pilha do AWS CloudFormation implantado anteriormente (em vez de criar uma nova).

- b. (Opcional) Insira uma das seguintes LogLevelconfigurações para a execução da função Lambda do seu aplicativo:
    - DEBUG
    - ERROR
    - INFO (padrão)
    - WARNING
5. Escolha I acknowledge that this app creates custom IAM roles, resource policies and deploys nested applications. (Eu reconheço que este aplicativo cria funções personalizadas do IAM,

políticas de recursos e implanta aplicativos aninhados.) e, na parte inferior da página, selecione Deploy (Implantar).

Na página Status de implantação para fork-example-ecommerce - *my-app*, o Lambda exibe o status Seu aplicativo está sendo implantado.

Na seção Resources (Recursos), o AWS CloudFormation começa a criar a pilha e exibe o status CREATE\_IN\_PROGRESS para cada recurso. Quando o processo é concluído, o AWS CloudFormation exibe o status CREATE\_COMPLETE.

#### Note

Pode levar entre 20–30 minutos para que todos os recursos sejam implantados.

Quando a implantação for concluída, o Lambda exibirá o status Your application has been deployed (Sua aplicação foi implantada).

## Etapa 2: Como executar o aplicativo de exemplo

1. No console do AWS Lambda, no painel de navegação, escolha Applications (Aplicativos).
2. Na página Applications (Aplicativos), no campo de pesquisa, busque `serverlessrepo-fork-example-ecommerce-my-app` e escolha o aplicativo.
3. Na seção Resources (Recursos), faça o seguinte:
  - a. Para encontrar o recurso cujo tipo é `ApiGatewayRestApi`, classifique os recursos por Tipo, por exemplo `ServerlessRestApi`, e expanda o recurso.
  - b. Dois recursos aninhados são exibidos, dos tipos `ApiGatewayDeployment` e `ApiGatewayStage`.
  - c. Copie o link Prod API endpoint (Endpoint da API de produção) e anexe `/checkout` a ele, por exemplo:

```
https://abcdefghijkl.execute-api.us-east-2.amazonaws.com/Prod/checkout
```

4. Copie o JSON a seguir em um arquivo denominado `test_event.json`.

```
{  
  "id": 15311,
```

```
"date": "2019-03-25T23:41:11-08:00",
"status": "confirmed",
"customer": {
  "id": 65144,
  "name": "John Doe",
  "email": "john.doe@example.com"
},
"payment": {
  "id": 2509,
  "amount": 450.00,
  "currency": "usd",
  "method": "credit",
  "card-network": "visa",
  "card-number": "1234 5678 9012 3456",
  "card-expiry": "10/2022",
  "card-owner": "John Doe",
  "card-cvv": "123"
},
"shipping": {
  "id": 7600,
  "time": 2,
  "unit": "days",
  "method": "courier"
},
"items": [{
  "id": 6512,
  "product": 8711,
  "name": "Hockey Jersey - Large",
  "quantity": 1,
  "price": 400.00,
  "subtotal": 400.00
}, {
  "id": 9954,
  "product": 7600,
  "name": "Hockey Puck",
  "quantity": 2,
  "price": 25.00,
  "subtotal": 50.00
}]
}
```

5. Para enviar uma solicitação HTTPS ao endpoint da API, envie a carga do evento de exemplo como entrada, executando um comando `curl`, por exemplo:

```
curl -d "$(cat test_event.json)" https://abcdefghijkl.execute-api.us-east-2.amazonaws.com/Prod/checkout
```

A API retorna a seguinte resposta vazia, indicando uma execução bem-sucedida:

```
{ }
```

### Etapa 3: Como verificar a execução do aplicativo de exemplo e seus pipelines

#### Etapa 1: Como verificar a execução do pipeline de checkout de exemplo

1. Faça login no [console do Amazon DynamoDB](#).
2. No painel de navegação, escolha Tables (Tabelas).
3. Busque `serverlessrepo-fork-example` e escolha CheckoutTable.
4. Na página de detalhes da tabela, escolha Items (Itens) e selecione o item criado.

Os atributos armazenados são exibidos.

#### Etapa 2: Como verificar a execução do pipeline de armazenamento e backup de eventos

1. Faça login no [console do Amazon S3](#).
2. No painel de navegação, selecione Buckets.
3. Busque `serverlessrepo-fork-example` e escolha CheckoutBucket.
4. Navegue pela hierarquia de diretórios até encontrar um arquivo com a extensão `.gz`.
5. Para fazer download do arquivo, escolha Actions (Ações), Open (Abrir).
6. O pipeline está configurado com uma função do Lambda que limpa as informações de cartão de crédito por motivos de conformidade.

Para verificar se a carga JSON armazenada não contém informações de cartão de crédito, descompacte o arquivo.

#### Etapa 3: Como verificar a execução do pipeline de pesquisa e análise de eventos

1. Faça login no [console OpenSearch de serviço](#).

2. No painel de navegação, em My domains (Meus domínios), escolha o domínio prefixado com `server1-analyt`.
3. O pipeline está configurado com uma política de filtro de assinatura do Amazon SNS que define uma condição de correspondência numérica.

Para verificar se o evento está indexado pois se refere a um pedido cujo valor é maior que 100 USD, na página `server1-analyt-abcdefgh1ijk`, escolha Índices (Índices), `checkout_events`.

#### Etapa 4: Como verificar a execução do pipeline de repetição de eventos

1. Faça login no [console do Amazon SQS](#).
2. Na lista de filas, busque `serverlessrepo-fork-example` e escolha `ReplayQueue`.
3. Escolha `Send and receive messages` (Enviar e receber mensagens).
4. Na caixa de diálogo `Enviar e receber mensagens em fork-example-ecommerce - my-app... replayP- ReplayQueue - 123ABCD4E5F6`, escolha `Sondagem de mensagens`.
5. Para verificar se o evento está enfileirado, escolha `More Details` (Mais detalhes) ao lado da mensagem que aparece na fila.

#### Etapa 4: Como simular um problema e repetir eventos para recuperação

##### Etapa 1: Como habilitar o problema simulado e enviar uma segunda solicitação da API

1. Faça login no [AWS Lambdaconsole](#).
2. No painel de navegação, escolha `Functions` (Funções).
3. Busque `serverlessrepo-fork-example` e escolha `CheckoutFunction`.
4. **No `fork-example-ecommerce-my-app - - CheckoutFunction ABCDEF...` página, na seção Variáveis de ambiente, defina a variável `BUG_ENABLED` como verdadeira e escolha `Salvar`.**
5. Copie o JSON a seguir em um arquivo denominado `test_event_2.json`.

```
{
  "id": 9917,
  "date": "2019-03-26T21:11:10-08:00",
  "status": "confirmed",
  "customer": {
    "id": 56999,
```

```
    "name": "Marcia Oliveira",
    "email": "marcia.oliveira@example.com"
  },
  "payment": {
    "id": 3311,
    "amount": 75.00,
    "currency": "usd",
    "method": "credit",
    "card-network": "mastercard",
    "card-number": "1234 5678 9012 3456",
    "card-expiry": "12/2025",
    "card-owner": "Marcia Oliveira",
    "card-cvv": "321"
  },
  "shipping": {
    "id": 9900,
    "time": 20,
    "unit": "days",
    "method": "plane"
  },
  "items": [{
    "id": 9993,
    "product": 3120,
    "name": "Hockey Stick",
    "quantity": 1,
    "price": 75.00,
    "subtotal": 75.00
  }]
}
```

6. Para enviar uma solicitação HTTPS ao endpoint da API, envie a carga do evento de exemplo como entrada, executando um comando `curl`, por exemplo:

```
curl -d "$(cat test_event_2.json)" https://abcdefghijkl.execute-api.us-east-2.amazonaws.com/Prod/checkout
```

A API retorna a seguinte resposta vazia, indicando uma execução bem-sucedida:

```
{ }
```

## Etapa 2: Como verificar a corrupção de dados simulados

1. Faça login no [console do Amazon DynamoDB](#).
2. No painel de navegação, escolha Tables (Tabelas).
3. Busque `serverlessrepo-fork-example` e escolha `CheckoutTable`.
4. Na página de detalhes da tabela, escolha Items (Itens) e selecione o item criado.

Os atributos armazenados são exibidos, alguns marcados como CORRUPTED! (CORROMPIDOS!)

## Etapa 3: Como desabilitar o problema simulado

1. Faça login no [AWS Lambdaconsole](#).
2. No painel de navegação, escolha Functions (Funções).
3. Busque `serverlessrepo-fork-example` e escolha `CheckoutFunction`.
4. ***No `fork-example-ecommerce-my-app - - CheckoutFunction ABCDEF...` página, na seção Variáveis de ambiente, defina a variável `BUG_ENABLED` como falsa e escolha Salvar.***

## Etapa 4: Como habilitar a repetição a fim de recuperar do problema

1. No console do AWS Lambda, no painel de navegação, escolha Functions (Funções).
2. Busque `serverlessrepo-fork-example` e escolha `ReplayFunction`.
3. Expanda a seção Designer, escolha o bloco SQS e, na seção SQS, selecione Enabled (Habilitado).

### Note

Leva aproximadamente 1 minuto para que o acionador da origem de eventos do Amazon SQS seja habilitado.

4. Escolha Salvar.
5. Para visualizar os atributos recuperados, volte ao console do Amazon DynamoDB.
6. Para desativar as novas tentativas, volte ao console do AWS Lambda e desabilite o acionador da origem de eventos do Amazon SQS para `ReplayFunction`.

## Inscrever um AWS Event Fork Pipelines em um tópico do Amazon SNS

Para acelerar o desenvolvimento de suas aplicações orientadas a eventos, você pode inscrever pipelines de tratamento de eventos, com a tecnologia AWS Event Fork Pipelines, em tópicos do Amazon SNS. AWS Event Fork Pipelines é um conjunto de [aplicativos aninhados](#) de código aberto, com base no [AWS Serverless Application Model](#) (AWS SAM), que você pode implantar diretamente do pacote [AWS Event Fork Pipelines](#) (escolha Show apps that create custom IAM roles or resource policies (Mostrar aplicativos que criam funções personalizadas do IAM ou políticas de recursos) em sua conta da AWS. Para ter mais informações, consulte [Como o AWS Event Fork Pipelines funciona](#).

Esta seção mostra como usar o AWS Management Console para implantar um pipeline e, em seguida, inscrever o AWS Event Fork Pipelines em um tópico do Amazon SNS. Antes de começar, [crie um tópico do Amazon SNS](#).

Para excluir os recursos que compõem um pipeline, localize o pipeline na página Aplicativos do AWS Lambda console, expanda a seção de modelos do SAM, escolha CloudFormation pilha e, em seguida, escolha Outras ações, Excluir pilha.

### Tópicos

- [Como implantar e inscrever o pipeline de armazenamento e backup de eventos](#)
- [Como implantar e inscrever o pipeline de pesquisa e análise de eventos](#)
- [Como implantar e inscrever o pipeline de repetição de eventos](#)

## Como implantar e inscrever o pipeline de armazenamento e backup de eventos

Para arquivamento e análise de eventos, o Amazon SNS agora recomenda usar sua integração nativa com o Amazon Data Firehose. Você pode inscrever streams de entrega do Firehose em tópicos do SNS, o que permite enviar notificações para endpoints de arquivamento e análise, como buckets do Amazon Simple Storage Service (Amazon S3), tabelas do Amazon Redshift, Amazon Service (Service) e muito mais. OpenSearch Usar o Amazon SNS com os streams de entrega do Firehose é uma solução totalmente gerenciada e sem código que não exige o uso de funções. AWS Lambda Para ter mais informações, consulte [Fanout para streams de entrega do Firehose](#).

Este tutorial mostra como implantar o [pipeline de armazenamento e backup de eventos](#) e inscrevê-lo em um tópico do Amazon SNS. Esse processo transforma automaticamente o modelo do AWS

SAM associado ao pipeline em uma pilha do AWS CloudFormation e implanta a pilha em sua Conta da AWS. Esse processo também cria e configura o conjunto de recursos que compõe o pipeline de armazenamento e backup de eventos, incluindo o seguinte:

- Fila do Amazon SQS
- Função do Lambda
- Fluxo de entrega do Firehose
- Bucket de backup do Amazon S3

Para obter mais informações sobre como configurar um stream com um bucket do S3 como destino, consulte a Amazon [S3DestinationConfiguration](#) Data Firehose API Reference.

Para obter mais informações sobre como transformar eventos e configurar o buffer de eventos, a compactação de eventos e a criptografia de eventos, consulte Criação de [um stream de entrega do Amazon Data Firehose no Guia do desenvolvedor do Amazon Data Firehose](#).

Para obter mais informações sobre como filtrar eventos, consulte [Políticas de filtro de assinatura do Amazon SNS](#) neste guia.

1. Faça login no [AWS Lambdaconsole](#).
2. No painel de navegação, escolha Functions (Funções) e selecione Create function (Criar função).
3. Na página Create function (Criar função), faça o seguinte:
  - a. Escolha Browse serverless app repository (Procurar no repositório de aplicações sem servidor), Public applications (Aplicações públicas), Show apps that create custom IAM roles or resource policies (Mostrar aplicações que criam funções do IAM ou políticas de recursos).
  - b. Procure `fork-event-storage-backup-pipeline` e escolha o aplicativo.
4. Na página `fork-event-storage-backup-pipeline`, faça o seguinte:
  - a. Na seção Application settings (Configurações do aplicativo), insira um Application name (Nome de aplicativo) (por exemplo, `my-app-backup`).

 Note

- Para cada implantação, o nome do aplicativo deve ser exclusivo. Se você reutilizar um nome de aplicativo, a implantação atualizará apenas a pilha do AWS CloudFormation implantado anteriormente (em vez de criar uma nova).

- (Opcional) Para `BucketArn`, insira o ARN do bucket do S3 no qual os eventos de entrada são carregados. Se você não inserir um valor, um novo bucket do S3 será criado em sua conta da AWS.
- (Opcional) Para `DataTransformationFunctionArn`, insira o ARN da função Lambda por meio da qual os eventos recebidos são transformados. Se você não inserir um valor, a transformação de dados será desativada.
- (Opcional) Insira uma das seguintes `LogLevel` configurações para a execução da função Lambda do seu aplicativo:
  - DEBUG
  - ERROR
  - INFO (padrão)
  - WARNING
- Para `TopicArn`, insira o ARN do tópico do Amazon SNS no qual essa instância do fork pipeline deve ser assinada.
- (Opcional) Para `StreamBufferingIntervalInSeconds` e `StreamBufferingSizeInMBs`, insira os valores para configurar o armazenamento em buffer de eventos recebidos. Se nenhum valor for inserido, são usados 300 segundos e 5 MB.
- (Opcional) Insira uma das seguintes `StreamCompressionFormat` configurações para compactar eventos recebidos:
  - GZIP
  - SNAPPY
  - UNCOMPRESSED (padrão)
  - ZIP
- (Opcional) Para `StreamPrefix`, insira o prefixo da string para nomear os arquivos armazenados no bucket de backup do S3. Se você não inserir um valor, nenhum prefixo será usado.

- i. (Opcional) Para `SubscriptionFilterPolicy`, insira a política de filtro de assinatura do Amazon SNS, no formato JSON, a ser usada para filtrar eventos recebidos. A política de filtro decide quais eventos são indexados no índice OpenSearch de serviços. Se nenhum valor for inserido, nenhuma filtragem será usada (todos os eventos serão indexados).
- j. (Opcional) Para `SubscriptionFilterPolicyScope`, insira a string `MessageBody` ou habilite `MessageAttributes` a filtragem de mensagens baseada em carga ou atributo.
- k. Escolha `I acknowledge that this app creates custom IAM roles, resource policies and deploys nested applications.` (Eu reconheço que este aplicativo cria funções personalizadas do IAM, políticas de recursos e implanta aplicativos aninhados.) e selecione `Deploy` (Implantar).

Na página `Deployment status for my-app` (Status de implantação para *my-app*), o Lambda exibe o status `Your application is being deployed` (Sua aplicação está sendo implantada).

Na seção `Resources` (Recursos), o AWS CloudFormation começa a criar a pilha e exibe o status `CREATE_IN_PROGRESS` para cada recurso. Quando o processo é concluído, o AWS CloudFormation exibe o status `CREATE_COMPLETE`.

Quando a implantação for concluída, o Lambda exibirá o status `Your application has been deployed` (Sua aplicação foi implantada).

As mensagens publicadas em seu tópico do Amazon SNS são armazenadas no bucket de backup do S3 provisionado pelo pipeline de armazenamento e backup de eventos automaticamente.

## Como implantar e inscrever o pipeline de pesquisa e análise de eventos

Para arquivamento e análise de eventos, o Amazon SNS agora recomenda usar sua integração nativa com o Amazon Data Firehose. Você pode inscrever streams de entrega do Firehose em tópicos do SNS, o que permite enviar notificações para endpoints de arquivamento e análise, como buckets do Amazon Simple Storage Service (Amazon S3), tabelas do Amazon Redshift, Amazon Service (Service) e muito mais. OpenSearch OpenSearch Usar o Amazon SNS com os streams de entrega do Firehose é uma solução totalmente gerenciada e sem código que não exige o uso de funções. AWS Lambda Para ter mais informações, consulte [Fanout para streams de entrega do Firehose](#).

Esta página mostra como implantar o [pipeline de pesquisa e análise de eventos](#) e inscrevê-lo em um tópico do Amazon SNS. Esse processo transforma automaticamente o modelo do AWS SAM associado ao pipeline em uma pilha do AWS CloudFormation e implanta a pilha em sua Conta da AWS. Esse processo também cria e configura o conjunto de recursos que compõe o pipeline de pesquisa e análise de eventos, incluindo o seguinte:

- Fila do Amazon SQS
- Função do Lambda
- Fluxo de entrega do Firehose
- Domínio do Amazon OpenSearch Service
- Bucket de mensagens mortas do Amazon S3

Para obter mais informações sobre como configurar um stream com um índice como destino, consulte [ElasticsearchDestinationConfiguration](#) a Amazon Data Firehose API Reference.

Para obter mais informações sobre como transformar eventos e configurar o buffer de eventos, a compactação de eventos e a criptografia de eventos, consulte Criação de [um stream de entrega do Amazon Data Firehose no Guia do desenvolvedor do Amazon Data Firehose](#).

Para obter mais informações sobre como filtrar eventos, consulte [Políticas de filtro de assinatura do Amazon SNS](#) neste guia.

1. Faça login no [AWS Lambdaconsole](#).
2. No painel de navegação, escolha Functions (Funções) e selecione Create function (Criar função).
3. Na página Create function (Criar função), faça o seguinte:
  - a. Escolha Browse serverless app repository (Procurar no repositório de aplicações sem servidor), Public applications (Aplicações públicas), Show apps that create custom IAM roles or resource policies (Mostrar aplicações que criam funções do IAM ou políticas de recursos).
  - b. Procure `fork-event-search-analytics-pipeline` e escolha o aplicativo.
4. Na página `fork-event-search-analytics-pipeline`, faça o seguinte:
  - a. Na seção Application settings (Configurações do aplicativo), insira um Application name (Nome de aplicativo) (por exemplo, `my-app-search`).

**Note**

Para cada implantação, o nome do aplicativo deve ser exclusivo. Se você reutilizar um nome de aplicativo, a implantação atualizará apenas a pilha do AWS CloudFormation implantado anteriormente (em vez de criar uma nova).

- b. (Opcional) Para `DataTransformationFunctionArn`, insira o ARN da função Lambda usada para transformar eventos recebidos. Se você não inserir um valor, a transformação de dados será desativada.
- c. (Opcional) Insira uma das seguintes `LogLevel` configurações para a execução da função Lambda do seu aplicativo:
  - DEBUG
  - ERROR
  - INFO (padrão)
  - WARNING
- d. (Opcional) Em `SearchDomainArn`, insira o ARN do domínio de OpenSearch serviço, um cluster que configura a funcionalidade necessária de computação e armazenamento. Se você não inserir um valor, um novo domínio será criado com a configuração padrão.
- e. Para `TopicArn`, insira o ARN do tópico do Amazon SNS no qual essa instância do fork pipeline deve ser assinada.
- f. Para `SearchIndexName`, insira o nome do índice de OpenSearch serviços para pesquisa e análise de eventos.

**Note**

As seguintes cotas se aplicam a nomes de índice:

- Não é possível incluir letras maiúsculas
- Não é possível incluir os seguintes caracteres: `\ / * ? " < > | ` , #`
- Não é possível começar com os seguintes caracteres: `- + _`
- Não podem ser o seguinte: `. . .`
- Não podem ter mais que 80 caracteres
- Não podem ter mais que 255 bytes

- Não é possível conter dois pontos (do OpenSearch Serviço 7.0)

g. (Opcional) Insira uma das seguintes `SearchIndexRotationPeriod` configurações para o período de rotação do índice de OpenSearch serviços:

- `NoRotation` (padrão)
- `OneDay`
- `OneHour`
- `OneMonth`
- `OneWeek`

A rotação de índice anexa um timestamp ao nome do índice, facilitando a expiração de dados antigos.

h. Para `SearchTypeName`, insira o nome do tipo de OpenSearch serviço para organizar os eventos em um índice.

 Note

- OpenSearch Os nomes dos tipos de serviço podem conter qualquer caractere (exceto bytes nulos), mas não podem começar com `_` eles.
- Para o OpenSearch Service 6.x, só pode haver um tipo por índice. Se você especificar um novo tipo para um índice existente que já tenha outro tipo, o Firehose retornará um erro de tempo de execução.

i. (Opcional) Para `StreamBufferingIntervalInSeconds` `StreamBufferingSizeInMBs`, insira os valores para configurar o armazenamento em buffer de eventos recebidos. Se nenhum valor for inserido, são usados 300 segundos e 5 MB.

j. (Opcional) Insira uma das seguintes `StreamCompressionFormat` configurações para compactar eventos recebidos:

- `GZIP`
- `SNAPPY`
- `UNCOMPRESSED` (padrão)
- `ZIP`

- k. (Opcional) Para `StreamPrefix`, insira o prefixo da string para nomear os arquivos armazenados no bucket de letras mortas do S3. Se você não inserir um valor, nenhum prefixo será usado.
- l. (Opcional) Para `StreamRetryDurationInSeconds`, insira a duração da nova tentativa para casos em que o Firehose não consegue indexar eventos OpenSearch no índice de serviços. Se você não inserir um valor, será usado 300 segundos.
- m. (Opcional) Para `SubscriptionFilterPolicy`, insira a política de filtro de assinatura do Amazon SNS, no formato JSON, a ser usada para filtrar eventos recebidos. A política de filtro decide quais eventos são indexados no índice OpenSearch de serviços. Se nenhum valor for inserido, nenhuma filtragem será usada (todos os eventos serão indexados).
- n. Escolha `I acknowledge that this app creates custom IAM roles, resource policies and deploys nested applications.` (Eu reconheço que este aplicativo cria funções personalizadas do IAM, políticas de recursos e implanta aplicativos aninhados.) e selecione `Deploy` (Implantar).

Na *my-app-search* página Status de implantação para, o Lambda exibe o status Seu aplicativo está sendo implantado.

Na seção Resources (Recursos), o AWS CloudFormation começa a criar a pilha e exibe o status `CREATE_IN_PROGRESS` para cada recurso. Quando o processo é concluído, o AWS CloudFormation exibe o status `CREATE_COMPLETE`.

Quando a implantação for concluída, o Lambda exibirá o status `Your application has been deployed` (Sua aplicação foi implantada).

As mensagens publicadas em seu tópico do Amazon SNS são indexadas automaticamente no índice de OpenSearch serviços provisionado pelo pipeline de pesquisa e análise de eventos. Se o pipeline não puder indexar um evento, ele armazenará em um bucket de mensagens mortas do S3.

## Como implantar e inscrever o pipeline de repetição de eventos

Esta página mostra como implantar o [pipeline de repetição de eventos](#) e inscrevê-lo em um tópico do Amazon SNS. Esse processo transforma automaticamente o modelo do AWS SAM associado ao pipeline em uma pilha do AWS CloudFormation e implanta a pilha em sua Conta da AWS. Esse processo também cria e configura o conjunto de recursos que compõe o pipeline de repetição de eventos, incluindo uma fila do Amazon SQS e uma função do Lambda.

Para obter mais informações sobre como filtrar eventos, consulte [Políticas de filtro de assinatura do Amazon SNS](#) neste guia.

1. Faça login no [AWS Lambdaconsole](#).
2. No painel de navegação, escolha Functions (Funções) e selecione Create function (Criar função).
3. Na página Create function (Criar função), faça o seguinte:
  - a. Escolha Browse serverless app repository (Procurar no repositório de aplicações sem servidor), Public applications (Aplicações públicas), Show apps that create custom IAM roles or resource policies (Mostrar aplicações que criam funções do IAM ou políticas de recursos).
  - b. Procure `fork-event-replay-pipeline` e escolha o aplicativo.
4. Na página `fork-event-replay-pipeline`, faça o seguinte:
  - a. Na seção Application settings (Configurações do aplicativo), insira um Application name (Nome de aplicativo) (por exemplo, `my-app-replay`).

 Note

Para cada implantação, o nome do aplicativo deve ser exclusivo. Se você reutilizar um nome de aplicativo, a implantação atualizará apenas a pilha do AWS CloudFormation implantado anteriormente (em vez de criar uma nova).

- b. (Opcional) Insira uma das seguintes LogLevelconfigurações para a execução da função Lambda do seu aplicativo:
  - DEBUG
  - ERROR
  - INFO (padrão)
  - WARNING
- c. (Opcional) Para `ReplayQueueRetentionPeriodInSeconds`, insira o tempo, em segundos, durante o qual a fila de repetição do Amazon SQS mantém a mensagem. Se você não inserir um valor, será usado 1.209.600 segundos (14 dias).
- d. Para `TopicArn`, insira o ARN do tópico do Amazon SNS no qual essa instância do fork pipeline deve ser assinada.

- e. Para `DestinationQueueName`, insira o nome da fila do Amazon SQS para a qual a função de reprodução do Lambda encaminha as mensagens.
- f. (Opcional) Para `SubscriptionFilterPolicy`, insira a política de filtro de assinatura do Amazon SNS, no formato JSON, a ser usada para filtrar eventos recebidos. A política de filtro decide quais eventos serão armazenados em buffer para repetição. Se nenhum valor for inserido, nenhuma filtragem será usada (todos os eventos serão armazenados em buffer).
- g. Escolha `I acknowledge that this app creates custom IAM roles, resource policies and deploys nested applications.` (Eu reconheço que este aplicativo cria funções personalizadas do IAM, políticas de recursos e implanta aplicativos aninhados.) e selecione `Deploy` (Implantar).

Na `my-app-replay` página Status de implantação para, o Lambda exibe o status Seu aplicativo está sendo implantado.

Na seção Resources (Recursos), o AWS CloudFormation começa a criar a pilha e exibe o status `CREATE_IN_PROGRESS` para cada recurso. Quando o processo é concluído, o AWS CloudFormation exibe o status `CREATE_COMPLETE`.

Quando a implantação for concluída, o Lambda exibirá o status `Your application has been deployed` (Sua aplicação foi implantada).

As mensagens publicadas em seu tópico do Amazon SNS são armazenadas em buffer para repetição na fila do Amazon SQS provisionada pelo pipeline de repetição de eventos automaticamente.

#### Note

Por padrão, a repetição está desabilitada. Para habilitar a repetição, navegue até a página da função no console do Lambda, expanda a seção Designer, escolha o bloco SQS e, na seção SQS, escolha `Enabled` (Habilitado).

## Usar o Agendador do Amazon EventBridge com o Amazon SNS

O [Agendador do Amazon EventBridge](#) é um agendador sem servidor que permite criar, executar e gerenciar tarefas de um serviço gerenciado central. Com o Agendador do EventBridge, é possível criar programações usando expressões cron e rate para padrões recorrentes ou configurar

invocações únicas. É possível configurar janelas de tempo flexíveis para entrega, bem como definir limites de repetição e o tempo máximo de retenção para invocações de API com falha.

Esta página explica como usar o Agendador do EventBridge para publicar uma mensagem de um tópico do Amazon SNS em uma programação.

## Tópicos

- [Configurar o perfil de execução](#)
- [Criar uma programação](#)
- [Recursos relacionados](#)

## Configurar o perfil de execução

Quando você cria uma nova programação, o Agendador EventBridge deve ter permissão para invocar sua operação de API de destino em seu nome. Você concede essas permissões ao Agendador EventBridge usando um perfil de execução. A política de permissão que você anexa ao perfil de execução da sua programação define as permissões necessárias. Essas permissões dependem da API de destino que você deseja que o Agendador EventBridge invoque.

Quando você usa o console do Agendador EventBridge para criar uma programação, como no procedimento a seguir, o Agendador EventBridge configura automaticamente um perfil de execução com base no destino selecionado. Se você quiser criar uma agenda usando um dos SDKs do Agendador EventBridge, a AWS CLI ou o AWS CloudFormation, você deve ter um perfil de execução existente que conceda as permissões que o Agendador EventBridge exige para invocar um alvo. Para obter mais informações sobre como configurar manualmente um perfil de execução para sua programação, consulte [Setting up an execution role](#) no Guia do usuário do Agendador EventBridge.

## Criar uma programação

Criar uma programação usando o console

1. Abra o console do Agendador do Amazon EventBridge em <https://console.aws.amazon.com/scheduler/home>.
2. Na página Programações, escolha Criar uma programação.
3. Na página Especificar detalhes da programação, na seção Nome e descrição da programação, faça o seguinte:

- a. Em Nome da programação, insira um nome para a programação. Por exemplo, **MyTestSchedule**.
- b. (Opcional) Em Descrição, insira uma descrição para a sua programação. Por exemplo, **My first schedule**.
- c. Para Grupo de programação, escolha um grupo de programação na lista suspensa. Se você não tiver um grupo, escolha padrão. Para criar um grupo de programação, escolha criar sua própria programação.

Você usa grupos de programação para adicionar tags a eles.

4. • Escolha suas opções de programação.

Ocorrência	Fazer isso...	
<p>Programação única</p> <p>Uma programação única invoca um destino somente uma vez na data e hora que você especificar.</p>	<p>Para Data e hora, faça o seguinte:</p> <ul style="list-style-type: none"> <li>• Insira uma data válida no formato YYYY/MM/DD .</li> <li>• Insira um carimbo de data e hora no formato 24 horas, hh:mm.</li> <li>• Para Fuso horário, escolha o fuso horário.</li> </ul>	
<p>Programação recorrente</p> <p>Uma programação recorrente invoca uma meta em uma taxa que você especifica usando uma expressão cron ou expressão de taxa.</p>	<p>a. Em Tipo de cronograma, siga um destes procedimentos:</p> <ul style="list-style-type: none"> <li>• Para usar uma expressão cron para definir o cronograma, escolha Cronograma baseado em cron e insira a expressão cron.</li> </ul>	

Ocorrência	Fazer isso...	
	<ul style="list-style-type: none"><li>• Para usar uma expressão <code>rate</code> para definir o cronograma, escolha <code>Cronogram</code> a com base em <code>rate</code> e insira a expressão <code>rate</code>.</li></ul> <p>Para obter mais informações sobre expressões <code>cron</code> e <code>rate</code>, consulte <a href="#">Schedule types on EventBridge Scheduler</a> no Guia do usuário do Agendador do Amazon EventBridge.</p> <p>b. Para Janela de tempo flexível, escolha <code>Desativado</code> para desativar a opção ou escolher uma das janelas de tempo predefinidas. Por exemplo, se você escolher 15 minutos e definir uma programação recorrente para invocar sua meta uma vez a cada hora, a programação será executada em até 15 minutos após o início de cada hora.</p>	

5. (Opcional) Se você escolher Programação recorrente na etapa anterior, na seção Período, faça o seguinte:
  - a. Para Fuso horário, escolha um fuso horário.
  - b. Para Data e hora de início, insira uma data válida no formato YYYY/MM/DD e, em seguida, especifique um carimbo de data e hora no formato 24 horas, hh : mm.
  - c. Para Data e hora de término, insira uma data válida no formato YYYY/MM/DD e, em seguida, especifique um carimbo de data e hora no formato 24 horas, hh : mm.
6. Escolha Next (Próximo).
7. Na página Selecionar destino, escolha a operação de API da AWS que o Agendador do EventBridge invoca:
  - a. Selecione Publicação do Amazon SNS.
  - b. Na seção Publicar, selecione um tópico do SNS ou escolha Criar novo tópico do SNS.
  - c. (Opcional) Insira uma carga útil JSON. Se você não inserir uma carga útil, o Agendador do EventBridge usará um evento vazio para invocar a função.
8. Escolha Next (Próximo).
9. Na página Settings (Configurações), faça o seguinte:
  - a. Para ativar a programação, em Estado da programação, alterne para Ativar programação.
  - b. Para configurar uma política de novas tentativas para a sua programação, em Política de novas tentativas e fila de mensagens não entregues (DLQ), faça o seguinte:
    - Alterne para Tentar novamente.
    - Para Idade máxima do evento, insira as horas e os minutos máximos que o Agendador do EventBridge deve manter um evento não processado.
    - O tempo máximo é de 24 horas.
    - Para Máximo de tentativas, insira o número máximo de vezes que o Agendador do EventBridge tentará novamente a programação se o destino retornar um erro.

O valor máximo é 185 novas tentativas.

Com políticas de novas tentativas, se uma programação falhar em invocar seu destino, o Agendador do EventBridge executará novamente a programação. Se configurado, você deve definir o tempo máximo de retenção e as novas tentativas para a programação.

- c. Escolha onde o Agendador do EventBridge armazena os eventos não entregues.

Opção Filas de mensagens não entregues (DLQ)	Fazer isso...	
Não armazene	Selecione None.	
Armazenar o evento na mesma Conta da AWS em que você está criando a programação	<p>a. Escolha Selecione uma fila do Amazon SQS na minha Conta da AWS como uma DLQ.</p> <p>b. Escolha o nome do recurso da Amazon (ARN) da fila do Amazon SQS.</p>	
Armazenar o evento em uma Conta da AWS diferente de onde você está criando a programação	<p>a. Escolha Especifique uma fila do Amazon SQS em outras Contas da AWS como uma DLQ.</p> <p>b. Insira o nome do recurso da Amazon (ARN) da fila do Amazon SQS.</p>	

- d. Para usar uma chave gerenciada pelo cliente para criptografar sua entrada de destino, em Criptografia, escolha Personalizar as configurações de criptografia (avançado).

Se você escolher essa opção, insira uma chave ARN do KMS existente ou escolha Criar um AWS KMS key para navegar até o console do AWS KMS. Para obter mais informações sobre como o Agendador do EventBridge criptografa seus dados em repouso, consulte [Encryption at rest](#) no Guia do usuário do Agendador do Amazon EventBridge.

- e. Para que o Agendador do EventBridge crie um novo perfil de execução para você, escolha Criar um novo perfil para esta programação. Depois, insira um nome para Nome do perfil. Se você escolher essa opção, o Agendador do EventBridge anexará as permissões necessárias para seu destino de exemplo ao perfil.

## 10. Escolha Next (Próximo).

11. Na página Revisar e criar uma programação, revise os detalhes da sua programação. Em cada seção, escolha Editar para voltar a essa etapa e editar seus detalhes.
12. Escolha Criar programação.

Você pode ver uma lista das suas programações novas e existentes na página Programações. Na coluna Status, verifique se sua nova programação está Ativada.

## Recursos relacionados

Para obter mais informações sobre o Agendador do EventBridge, consulte o seguinte:

- [Guia do usuário do Agendador do EventBridge](#)
- [Referência da API do Agendador do EventBridge](#)
- [Preços do Agendador do EventBridge](#)

# Usar o Amazon SNS para sistema de mensagens de aplicação para pessoa (A2P)

Esta seção fornece informações sobre o uso do Amazon SNS para notificações ao usuário usando assinantes, como aplicativos móveis, números de celulares e endereços de e-mail.

## Tópicos

- [Mensagens de texto em dispositivos móveis \(SMS\)](#)
- [Notificações por push para dispositivos móveis](#)
- [Notificações por e-mail](#)

## Mensagens de texto em dispositivos móveis (SMS)

Você pode usar o Amazon SNS para enviar mensagens de texto ou mensagens SMS para dispositivos habilitados para SMS. Você pode [enviar uma mensagem diretamente para um número de telefone](#), ou enviar [uma mensagem para vários números de telefone](#) de uma só vez inscrevendo esses números em um tópico e enviando sua mensagem para o tópico.

Você pode [definir as preferências de SMS](#) para sua conta da AWS para ajustar suas entregas de SMS de acordo com seus casos de uso e orçamento. Por exemplo, você pode escolher se as mensagens são otimizadas para custo ou entrega confiável. Você também pode especificar cotas de gastos para entregas de mensagens individuais e cotas de gastos mensais para sua Conta da AWS.

Quando exigido por leis e regulamentos locais (como nos Estados Unidos e no Canadá), os destinatários de SMS podem [optar pelo cancelamento](#), o que significa que eles podem optar por interromper o recebimento de mensagens SMS de sua Conta da AWS. Depois que um destinatário cancela o recebimento, você pode, com limitações, incluir o número de telefone para retomar o envio de mensagens para ele.

O Amazon SNS é compatível com sistemas de mensagens SMS em várias regiões, e você pode enviar mensagens para mais de 200 países e regiões. Para obter mais informações, consulte [Países e regiões compatíveis](#).

## Tópicos

- [Sandbox de SMS](#)

- [Identidades de origem para mensagens SMS](#)
- [Solicitar suporte para sistema de mensagens SMS com o Amazon SNS](#)
- [Definir preferências de mensagens SMS](#)
- [Enviar mensagens SMS](#)
- [Monitorar e atividade de SMS](#)
- [Gerenciar assinaturas de números de telefone e SMS](#)
- [Países e regiões compatíveis](#)
- [Práticas recomendadas para SMS](#)

## Sandbox de SMS

Ao começar a usar o Amazon SNS para enviar mensagens SMS, a conta da AWS estará na sandbox do SMS. A sandbox de SMS oferece um ambiente seguro para testar os recursos do Amazon SNS sem arriscar a reputação como remetente do SMS. Enquanto a conta estiver na sandbox de SMS, todos os recursos do Amazon SNS poderão ser utilizados, com as seguintes restrições:

- As mensagens SMS poderão ser enviadas apenas para números de telefone de destino verificados.
- É possível ter até dez números de telefone de destino verificados.
- A exclusão de números de telefone de destino só é possível após 24 horas ou mais desde a verificação ou desde a última tentativa de verificação.

Quando a conta for movida para fora da sandbox, essas restrições serão removidas e será possível enviar mensagens SMS para qualquer destinatário.

### Tópicos

- [Adicionar e verificar números de telefone na sandbox de SMS](#)
- [Excluir números de telefone da sandbox de SMS](#)
- [Sair da sandbox de SMS](#)

## Adicionar e verificar números de telefone na sandbox de SMS

Para começar a enviar mensagens SMS enquanto sua AWS conta está na [sandbox de SMS](#), crie uma [identidade de origem, adicione os números de](#) telefone de destino e, em seguida, verifique-os.

**Note**

Assim como acontece com contas que não estão na sandbox de SMS, uma [identidade de origem](#) é necessária antes de enviar mensagens SMS para destinatários em alguns países ou regiões. Para ter mais informações, consulte [Países e regiões compatíveis](#).

As IDs de origem incluem [IDs do remetente](#) e diferentes tipos de [números de origem](#). Para visualizar seus números de origem existentes, no painel de navegação do [console do Amazon SNS](#), escolha [Origination numbers \(Números de origem\)](#). Atualmente, os IDs de remetente não são exibidas nesta lista.

Para adicionar e verificar números de telefone de destino

1. Faça login no [console do Amazon SNS](#).
2. Crie uma [identidade de origem](#) para o número de telefone.
3. No menu do console, selecione uma [região da AWS que seja compatível com mensagens SMS](#).
4. No painel de navegação, escolha Mensagens de texto (SMS).
5. Na página Mobile text messaging (SMS) (Mensagens de texto móveis (SMS)), em Sandbox destination phone numbers (Números de telefone de destino da sandbox), escolha Add phone number (Adicionar número de telefone).
6. Em Destination details (Detalhes do destino), insira o código do país e o número do telefone, especifique o idioma a ser usado na mensagem de verificação e, em seguida, escolha Add phone number (Adicionar número de telefone).

O Amazon SNS envia uma One-Time Password (OTP – Senha de uso único) para o número de telefone de destino. Se o número de telefone de destino não receber a OTP dentro de 15 minutos, escolha Resend verification code (Reenvio de código de verificação). Você pode enviar a OTP para o mesmo número de telefone de destino até cinco vezes a cada 24 horas.

7. Na caixa Verification code (Código de verificação), insira a OTP enviada para o número de telefone de destino e escolha Verify phone number (Verificar número de telefone).

O número de telefone de destino e seu status de verificação são exibidos na seção Sandbox destination phone numbers (Números de telefone de destino da sandbox). Se o status da verificação for Pending (Pendente), a verificação não foi bem-sucedida. Isso poderá acontecer, por exemplo, se você não inseriu o código do país com o número de telefone. A exclusão de

números de telefone de destino com status pendente ou verificado só é possível após 24 horas ou mais desde a verificação ou desde a última tentativa de verificação.

8. Repita essas etapas em cada região em que esse número de telefone de destino deve ser utilizado.

## Solução de problemas do não recebimento de um texto OTP

Solucione problemas comuns que podem impedir que um número de telefone receba mensagens de texto OTP.

- Limite de gastos com SMS do Amazon SNS: se você Conta da AWS excedeu o limite de gastos para enviar mensagens SMS, outras mensagens, incluindo textos OTP, podem não ser entregues até que o limite seja aumentado ou o problema de cobrança seja resolvido.
- Número de telefone não ativado para notificações por SMS: em alguns países ou regiões, os destinatários devem optar por receber mensagens SMS a partir de códigos curtos, que são comumente usados para textos OTP. Se o número de telefone do destinatário não estiver ativado, ele não receberá o texto OTP.
- Restrições ou filtragem da operadora: algumas operadoras de celular podem ter restrições ou mecanismos de filtragem que impedem a entrega de certos tipos de mensagens SMS, incluindo textos OTP. Isso pode ser devido a políticas de segurança ou medidas anti-spam implementadas pela operadora.
- Número de telefone inválido ou incorreto: se o número de telefone fornecido pelo destinatário estiver incorreto ou inválido, o texto OTP não será entregue.
- Problemas de rede: problemas ou interrupções temporárias na rede podem impedir a entrega de mensagens SMS, incluindo textos OTP, ao telefone do destinatário.
- Entrega atrasada: em alguns casos, as mensagens SMS podem sofrer atrasos na entrega devido ao congestionamento da rede ou a outros fatores. O texto OTP pode eventualmente ser entregue, mas pode ser adiado além do prazo esperado.
- Suspensão ou encerramento da conta: Se houver problemas com você Conta da AWS, como falta de pagamento ou violação dos AWS termos de serviço, os recursos de mensagens do Amazon SNS, incluindo mensagens de texto OTP, podem ser suspensos ou encerrados.

## Excluir números de telefone da sandbox de SMS

É possível excluir números de telefone de destino pendentes ou verificados da [sandbox de SMS](#).

## Para excluir números de telefone de destino da sandbox de SMS

1. Aguarde 24 horas após [verificar o número de telefone](#) ou após a última tentativa de verificação.
2. Faça login no [console do Amazon SNS](#).
3. No menu do console, selecione uma [Região da AWS compatível com mensagens SMS](#) em que você adicionou um número de telefone de destino.
4. No painel de navegação, escolha Mensagens de texto (SMS).
5. Na página Mobile text messaging (SMS) (Mensagens de texto móveis (SMS)), em Sandbox destination phone numbers (Números de telefone de destino da sandbox), escolha o número de telefone a ser excluído e, em seguida, escolha Delete phone number (Excluir número de telefone).
6. Para confirmar que deseja excluir o número de telefone, insira **delete me** e escolha Delete (Excluir).

Se passaram 24 horas ou mais desde que você verificou ou tentou verificar o número de telefone de destino, ele será excluído e o Amazon SNS atualizará a lista de números de telefone de destino.

7. Repita essas etapas em todas as regiões em que esse número de telefone de destino deve ser utilizado.

## Sair da sandbox de SMS

Para Conta da AWS sair da [sandbox de SMS](#), você precisa primeiro adicionar, verificar e testar os números de telefone de destino. Em seguida, você deve criar um caso com AWS Support.

Para solicitar que sua AWS conta seja removida da sandbox de SMS

1. Verifique os números de telefone
  - a. Enquanto você Conta da AWS estiver na sandbox do SMS, abra o console do [Amazon SNS](#).
  - b. No painel de navegação, em Celular, escolha Mensagens de texto (SMS).
  - c. Na seção Números de telefone de destino do Sandbox, [adicione e verifique](#) um ou mais números de telefone de destino. Essa verificação garante que você possa enviar e receber mensagens com sucesso.
2. Teste a publicação de SMS

- Confirme se você consegue enviar e receber mensagens para pelo menos um número de telefone verificado. Para obter instruções mais detalhadas sobre como publicar mensagens SMS, consulte [Publicar em um telefone celular](#).
3. Iniciar a edição do sandbox
    - Na página Mobile text messaging (SMS) (Mensagens de texto móveis (SMS)) do console do Amazon SNS, em Account information (Informações da conta), escolha Exit SMS sandbox (Sair da sandbox de SMS). Essa ação redireciona você para o Amazon [Support Center](#) e cria automaticamente um caso de suporte com a opção de aumento da cota de serviço selecionada.
  4. Preencha o formulário
    - No formulário de suporte em Aumento da cota de serviço, faça o seguinte:
      - i. Escolha escolher SNS Text Messaging como serviço.
      - ii. Forneça o URL do site ou o nome do aplicativo a partir do qual você pretende enviar mensagens SMS.
      - iii. Especifique o tipo de mensagem que você enviará: Senha de uso único, promocional ou transacional.
      - iv. Escolha a Região da AWS a partir da qual você enviará mensagens SMS.
      - v. Liste os países ou regiões onde você planeja enviar mensagens SMS.
      - vi. Descreva como seus clientes optam por receber mensagens.
      - vii. Inclua qualquer modelo de mensagem que você pretenda usar.
  5. Especificar cota e região
    - Em Requests (Solicitações), faça o seguinte:
      - i. Escolha para Região da AWS onde você deseja mover seu Conta da AWS.
      - ii. Escolha Limites gerais para o tipo de recurso.
      - iii. Escolha Sair do SMS Sandbox para obter cota.
      - iv. (Opcional) Para solicitar aumentos adicionais ou outros ajustes, escolha Adicionar outra solicitação e especifique os detalhes necessários.
      - v. Em Novo valor da cota, insira o limite em USD que você está solicitando.
  6. Detalhes adicionais

- a. Na descrição do caso, forneça detalhes adicionais relevantes à sua solicitação.
  - b. Em Opções de contato, escolha seu idioma de contato preferido.
7. Envie a solicitação
- Escolha Enviar para enviar sua solicitação AWS Support.

A AWS Support equipe fornece uma resposta inicial à sua solicitação em 24 horas.

Para evitar que nossos sistemas sejam usados para enviar conteúdo indesejado ou malicioso, consideramos cuidadosamente cada solicitação. Se for possível, atenderemos à sua solicitação dentro desse período de 24 horas. No entanto, se for necessário solicitar mais informações a você, o tempo de resolução poderá ser mais longo.

Se o seu caso de uso não estiver alinhado com nossas políticas, talvez não seja possível atender à sua solicitação.

## Identidades de origem para mensagens SMS

Ao enviar mensagens SMS usando o Amazon SNS, é possível se identificar para seus destinatários usando os seguintes tipos de identidades de origem:

- [IDs de remetente](#)
- [Números de origem](#)

### Note

As mensagens SMS do Amazon SNS estão disponíveis nas regiões em que o Amazon Pinpoint não tem suporte no momento. Se você opera na Europa (Estocolmo), no Oriente Médio (Bahrein), na Europa (Paris), na América do Sul (São Paulo) ou no Oeste dos EUA (Norte da Califórnia), abra o console do Amazon Pinpoint na região Leste dos EUA (Norte da Virgínia) para registrar sua empresa e campanha de 10DLC, mas não solicite um número de 10DLC. Em vez disso, use o [console do Service Quotas da AWS](#) para criar um caso de aumento de limite de serviço ao solicitar o número 10DLC para essa região. Para saber mais sobre como solicitar identidades de origem, consulte [Solicitar suporte para sistema de mensagens SMS com o Amazon SNS](#).

## IDs de remetente

Um ID de remetente é um nome alfabético que identifica o remetente de uma mensagem SMS. Quando você envia uma mensagem SMS usando um ID de remetente, e o destinatário está em uma área que oferece suporte à autenticação de ID de remetente, o ID desse remetente é exibido no dispositivo do destinatário em lugar de um número de telefone. Um ID de remetente fornece aos destinatários de SMS mais informações sobre o remetente do que um número de telefone, código longo ou código simplificado.

IDs de remetente são compatíveis em vários países e regiões em todo o mundo. Em alguns lugares, se a sua empresa envia mensagens SMS para clientes individuais, você deve usar um ID de remetente pré-registrado em uma agência reguladora ou grupo do setor. Para obter uma lista completa de países e regiões que oferecem suporte ou requerem IDs de remetente, consulte [Países e regiões compatíveis](#).

Não há cobrança adicional pelo uso de IDs de remetente. No entanto, o suporte e os requisitos para autenticação de ID de remetente variam. Vários mercados importantes (incluindo Canadá, China e Estados Unidos) não oferecem suporte ao uso de IDs de remetente. Algumas áreas exigem que as empresas que enviam mensagens SMS para clientes individuais usem um ID de remetente pré-registrado junto a uma agência reguladora ou grupo do setor.

### Important

A AWS proíbe o [spoofing de SMS](#), caso em que o ID do remetente é usado para se passar por outra pessoa, empresa ou produto. Use apenas um ID de remetente que represente uma marca ou marca comercial que você possua.

## Vantagens

Os IDs de remetente fornecem mais informações ao destinatário sobre o remetente da mensagem. É mais fácil estabelecer a identidade de sua marca usando um ID de remetente do que usando um código simplificado ou longo. Não há cobrança adicional pelo uso de um ID de remetente.

## Desvantagens

A oferta de suporte e os requisitos para a autenticação do ID de remetente não são consistentes em todos os países ou regiões. Vários mercados importantes (incluindo Canadá, China e Estados

Unidos) não oferecem suporte ao ID de remetente. Em algumas áreas, os IDs de remetente precisam ser pré-aprovados por uma agência reguladora para que você possa usá-los.

### Registro de ID de remetente por país

Você precisa abrir um caso com o AWS Support para [registrar IDs de remetente para mensagens SMS](#). Depois de abrir o caso de suporte, a AWS compartilhará outros documentos necessários. Também é necessário fornecer as informações a seguir para o país apropriado em que você está registrando o ID do remetente.

Nome do país	Tipo de mensagem	Restrições e requisitos de formato	Requisitos de registro
Austrália (AU)	Transacional e promocional	<ul style="list-style-type: none"> <li>Alfanumérico</li> <li>Máximo de 11 caracteres</li> <li>Sem espaços</li> <li>Sem caracteres especiais</li> <li>O ID do remetente deve ser o nome da marca da empresa que envia o SMS</li> </ul>	<ul style="list-style-type: none"> <li>O ID do remetente que você deseja registrar</li> <li>Região da AWS da qual o usuário chamará a API/ serviço</li> <li>Company name (Nome da empresa)</li> <li>Endereço da empresa (incluindo o cidade, estado/província e CEP)</li> <li>País da empresa</li> <li>URL da empresa (link para a aplicação ou site da empresa)</li> <li>Volume mensal estimado</li> <li>Explicação do caso de uso e finalidade das mensagens</li> </ul>

Nome do país	Tipo de mensagem	Restrições e requisitos de formato	Requisitos de registro
			<ul style="list-style-type: none"><li>• Se não for óbvia, forneça uma explicação sobre a relação entre o nome da empresa e o ID de remetente</li><li>• Número da licença empresarial oficial da empresa ou número de IVA</li><li>• Modelos de mensagens que você planeja enviar</li><li>• Uma licença de registro empresarial. Os exemplos incluem, dentre outros:<ul style="list-style-type: none"><li>• <a href="#">Número comercial australiano (ABN)</a></li><li>• <a href="#">Número da empresa australiana (ACN)</a></li><li>• <a href="#">Número do organismo registrado australiano (ARBN)</a></li><li>• <a href="#">Número de corporação indígena (ICN)</a></li></ul></li></ul>

Nome do país	Tipo de mensagem	Restrições e requisitos de formato	Requisitos de registro
			<ul style="list-style-type: none"><li>• Uma <a href="#">carta de autorização (LOA)</a></li></ul>

Nome do país	Tipo de mensagem	Restrições e requisitos de formato	Requisitos de registro
Bielorrússia (BY)	Somente mensagens transacionais	<ul style="list-style-type: none"><li>• Alfanumérico</li><li>• Máximo de 11 caracteres</li><li>• Sem espaços</li><li>• Sem caracteres especiais</li><li>• O ID do remetente deve ser o nome da marca da empresa que envia o SMS</li></ul>	<ul style="list-style-type: none"><li>• O ID do remetente que você deseja registrar</li><li>• Região da AWS da qual o usuário chamará a API/ serviço</li><li>• Company name (Nome da empresa)</li><li>• Endereço da empresa (incluindo o cidade, estado/província e CEP)</li><li>• País da empresa</li><li>• Número de contato da empresa</li><li>• URL da empresa (link para a aplicação ou site da empresa)</li><li>• Volume mensal estimado</li><li>• Explicação do caso de uso e finalidade das mensagens</li><li>• Se não for óbvia, forneça uma explicação sobre a relação entre o nome da empresa e o ID de remetente</li></ul>

Nome do país	Tipo de mensagem	Restrições e requisitos de formato	Requisitos de registro
			<ul style="list-style-type: none"><li>• Número da licença empresarial oficial da empresa ou número de IVA</li><li>• Modelos de mensagens que você planeja enviar</li></ul>

Nome do país	Tipo de mensagem	Restrições e requisitos de formato	Requisitos de registro
<p>China (CN)</p> <p>A China não exige registro de ID de remetente, mas exige registro de conteúdo/modelo com assinatura de corpo anexada [por exemplo, (Amazon)].</p>	<p>As seguintes palavras-chave não são permitidas:</p> <ul style="list-style-type: none"> <li>• Falun Gong</li> <li>• SB</li> <li>• Praça Tiananmen</li> </ul> <p>Mensagens das seguintes categorias:</p> <ul style="list-style-type: none"> <li>• Cartões de crédito</li> <li>• Pagamentos digitais (incluindo criptomoedas)</li> <li>• URLs de tráfego fraudulento (phishing ou spam)</li> <li>• Jogos de aposta</li> <li>• Conteúdo impróprio (adulto, violência, drogas, álcool)</li> <li>• Empréstimos</li> <li>• Cirurgia plástica</li> <li>• Política</li> <li>• Religião</li> <li>• Comércio de ações</li> <li>• Moedas virtuais</li> </ul> <p>As assinaturas entre colchetes devem ser</p>	<ul style="list-style-type: none"> <li>• Máximo de 11 caracteres</li> <li>• Sem espaços</li> <li>• Sem caracteres especiais</li> </ul>	<ul style="list-style-type: none"> <li>• Company name (Nome da empresa)</li> <li>• Endereço da empresa</li> <li>• Estado/província</li> <li>• País</li> <li>• Número de telefone da empresa</li> <li>• Site da empresa</li> <li>• Volume mensal estimado</li> <li>• Tipo de mensagem: promocional/transacional</li> <li>• Explicação do caso de uso</li> <li>• Modelos que você deseja registrar (para garantir a conformidade e a entrega, o SMS enviado não poderá ser diferente dos modelos fornecidos). Por exemplo, [CompanyName] Sua senha de uso único é {OTP}. Esse código vai expirar em 10 minutos.</li> </ul>

Nome do país	Tipo de mensagem	Restrições e requisitos de formato	Requisitos de registro
	<p>anexadas ao corpo da mensagem SMS. Para que a entrega seja eficaz na China, é necessário registrar uma assinatura de mensagem com seu modelo de conteúdo de mensagem. Essa assinatura de mensagem deve ser anexada ao conteúdo de cada SMS enviado. Se você não anexar a assinatura registrada ao corpo da mensagem SMS, o SMS poderá ser bloqueado ou filtrado. As assinaturas devem ser anexadas ao corpo do SMS entre colchetes.</p>		<ul style="list-style-type: none"><li>• Confirmação de que você não enviará conteúdo promocional como tipos de mensagem transacional</li><li>• Assinatura da mensagem. Consulte <a href="#">Restrições e requisitos de formato de assinatura</a>.</li></ul>

Nome do país	Tipo de mensagem	Restrições e requisitos de formato	Requisitos de registro
Egito (EG)	Somente mensagens transacionais	<ul style="list-style-type: none"> <li>• Alfanumérico</li> <li>• Máximo de 11 caracteres</li> <li>• Sem espaços</li> <li>• Sem caracteres especiais</li> </ul>	<ul style="list-style-type: none"> <li>• O ID do remetente que você deseja registrar</li> <li>• Região da AWS da qual o usuário chamará a API/ serviço</li> <li>• Company name (Nome da empresa)</li> <li>• Endereço da empresa (incluindo o cidade, estado/província e CEP)</li> <li>• País da empresa</li> <li>• Número de contato da empresa</li> <li>• URL da empresa (link para a aplicação ou site da empresa)</li> <li>• Volume mensal estimado</li> <li>• Explicação do caso de uso e finalidade das mensagens</li> <li>• Se não for óbvia, forneça uma explicação sobre a relação entre o nome da empresa e o ID de remetente</li> </ul>

Nome do país	Tipo de mensagem	Restrições e requisitos de formato	Requisitos de registro
			<ul style="list-style-type: none"> <li>• Sua empresa tem uma entidade/escritório comercial no Egito? Ou é uma empresa não sediada no Egito que envia conteúdo para o Egito?</li> <li>• Confirmação por escrito de que o caso de uso abrange todas as mensagens a serem enviadas por esse ID de remetente</li> <li>• Se não for óbvia, forneça a relação entre o nome da empresa e o ID de remetente</li> <li>• Modelos de mensagens que você planeja enviar</li> </ul>
Índia (IN)	Somente mensagens transacionais	<ul style="list-style-type: none"> <li>• Alfanumérico</li> <li>• Máximo de 6 caracteres</li> <li>• Sem espaços</li> <li>• Sem caracteres especiais</li> </ul>	Consulte <a href="#">Requisitos para registro de ID de remetente da Índia</a> .

Nome do país	Tipo de mensagem	Restrições e requisitos de formato	Requisitos de registro
Jordânia (JO)	Somente mensagens transacionais	<ul style="list-style-type: none"><li>• Alfanumérico</li><li>• Máximo de 11 caracteres</li><li>• Sem espaços</li><li>• Sem caracteres especiais</li></ul>	<ul style="list-style-type: none"><li>• O ID do remetente que você deseja registrar</li><li>• Região da AWS da qual o usuário chamará a API/ serviço</li><li>• Company name (Nome da empresa)</li><li>• Endereço da empresa (incluindo o cidade, estado/província e CEP)</li><li>• País da empresa</li><li>• Número de contato da empresa</li><li>• URL da empresa (link para a aplicação ou site da empresa)</li><li>• Volume mensal estimado</li><li>• Explicação do caso de uso e finalidade das mensagens</li><li>• Se não for óbvia, forneça uma explicação sobre a relação entre o nome da empresa e o ID de remetente</li></ul>

Nome do país	Tipo de mensagem	Restrições e requisitos de formato	Requisitos de registro
			<ul style="list-style-type: none"><li>• Certificado de cadastro empresarial</li><li>• O tipo de mensagem que você planeja enviar (por exemplo, OTP, alertas)</li><li>• Confirmação por escrito de que o caso de uso descreve todas as mensagens a serem enviadas por esse ID de remetente</li><li>• Modelos de mensagens que você planeja enviar</li></ul>

Nome do país	Tipo de mensagem	Restrições e requisitos de formato	Requisitos de registro
Kuwait (KW)	Somente mensagens transacionais	<ul style="list-style-type: none"><li>• Alfanumérico</li><li>• Máximo de 11 caracteres</li><li>• Sem espaços</li><li>• Sem caracteres especiais</li></ul>	<ul style="list-style-type: none"><li>• O ID do remetente que você deseja registrar</li><li>• Região da AWS da qual o usuário chamará a API/ serviço</li><li>• Company name (Nome da empresa)</li><li>• Endereço da empresa (incluindo o cidade, estado/província e CEP)</li><li>• País da empresa</li><li>• Número de contato da empresa</li><li>• URL da empresa (link para a aplicação ou site da empresa)</li><li>• Volume mensal estimado</li><li>• Explicação do caso de uso e finalidade das mensagens</li><li>• Se não for óbvia, forneça uma explicação sobre a relação entre o nome da empresa e o ID de remetente</li></ul>

Nome do país	Tipo de mensagem	Restrições e requisitos de formato	Requisitos de registro
			<ul style="list-style-type: none"><li>• O tipo de mensagem que você planeja enviar (por exemplo, OTP, alertas)</li><li>• Confirmação por escrito de que o caso de uso descreve todas as mensagens a serem enviadas por esse ID de remetente</li><li>• Modelos de mensagens que você planeja enviar</li></ul>

Nome do país	Tipo de mensagem	Restrições e requisitos de formato	Requisitos de registro
Filipinas (PH)	Somente mensagens transacionais	<ul style="list-style-type: none"><li>• Alfanumérico</li><li>• Máximo de 11 caracteres</li><li>• Sem espaços</li><li>• Sem caracteres especiais</li></ul>	<ul style="list-style-type: none"><li>• O ID do remetente que você deseja registrar</li><li>• Região da AWS da qual o usuário chamará a API/ serviço</li><li>• Company name (Nome da empresa)</li><li>• Endereço da empresa (incluindo o cidade, estado/província e CEP)</li><li>• País da empresa</li><li>• Número de contato da empresa</li><li>• URL da empresa (link para a aplicação ou site da empresa)</li><li>• Volume mensal estimado</li><li>• Explicação do caso de uso e finalidade das mensagens</li><li>• Se não for óbvia, forneça uma explicação sobre a relação entre o nome da empresa e o ID de remetente</li></ul>

Nome do país	Tipo de mensagem	Restrições e requisitos de formato	Requisitos de registro
			<ul style="list-style-type: none"><li>• O tipo de mensagem que você planeja enviar (por exemplo, OTP, alertas)</li><li>• Confirmação por escrito de que o caso de uso descreve todas as mensagens a serem enviadas por esse ID de remetente</li><li>• Modelos de mensagens que você planeja enviar</li></ul>

Nome do país	Tipo de mensagem	Restrições e requisitos de formato	Requisitos de registro
Catar (QA)	Somente mensagens transacionais	<ul style="list-style-type: none"><li>• Alfanumérico</li><li>• Máximo de 11 caracteres</li><li>• Sem espaços</li><li>• Sem caracteres especiais</li></ul>	<ul style="list-style-type: none"><li>• O ID do remetente que você deseja registrar</li><li>• Região da AWS da qual o usuário chamará a API/ serviço</li><li>• Company name (Nome da empresa)</li><li>• Endereço da empresa (incluindo o cidade, estado/província e CEP)</li><li>• País da empresa</li><li>• Número de contato da empresa</li><li>• URL da empresa (link para a aplicação ou site da empresa)</li><li>• Volume mensal estimado</li><li>• Explicação do caso de uso e finalidade das mensagens</li><li>• Se não for óbvia, forneça uma explicação sobre a relação entre o nome da empresa e o ID de remetente</li></ul>

Nome do país	Tipo de mensagem	Restrições e requisitos de formato	Requisitos de registro
			<ul style="list-style-type: none"><li>• Tipo de SMS que será enviado</li><li>• (Transacional/promocional/OTP) Observe que somente mensagens transacionais/OTP podem ser usadas com IDs de remetente destinados ao Catar para estabelecer conformidade.</li><li>• Confirmação por escrito de que o caso de uso descreve todas as mensagens a serem enviadas por esse ID de remetente</li><li>• Modelos de mensagens que você planeja enviar</li></ul>

Nome do país	Tipo de mensagem	Restrições e requisitos de formato	Requisitos de registro
Rússia (RU)	Somente mensagens transacionais	<ul style="list-style-type: none"><li>• Alfanumérico</li><li>• Máximo de 11 caracteres</li><li>• Sem espaços</li><li>• Sem caracteres especiais</li></ul>	<ul style="list-style-type: none"><li>• O ID do remetente que você deseja registrar</li><li>• Região da AWS da qual o usuário chamará a API/ serviço</li><li>• Company name (Nome da empresa)</li><li>• Endereço da empresa (incluindo o cidade, estado/província e CEP)</li><li>• País da empresa</li><li>• Número de contato da empresa</li><li>• URL da empresa (link para a aplicação ou site da empresa)</li><li>• ID fiscal ou número de licença</li><li>• Certificado de cadastro empresarial</li><li>• Endereço de e-mail para contato</li><li>• Volume mensal estimado</li></ul>

Nome do país	Tipo de mensagem	Restrições e requisitos de formato	Requisitos de registro
			<ul style="list-style-type: none"><li>• Explicação do caso de uso e finalidade das mensagens</li><li>• Se não for óbvia, forneça uma explicação sobre a relação entre o nome da empresa e o ID de remetente</li><li>• Confirmação de que esse caso de uso se aplicará a todas as mensagens enviadas dessa conta para a Rússia</li><li>• Confirmação de que mensagens não transacionais serão enviadas usando outro ID de remetente</li><li>• Confirmação de que você concorda com a taxa mensal de USD 272: Sim/ Não</li><li>• Modelos de mensagens que você planeja enviar</li></ul>

Nome do país	Tipo de mensagem	Restrições e requisitos de formato	Requisitos de registro
Arábia Saudita (SA)	Cada ID de remetente deve ser transaccional ou promocional. Um ID de remetente não pode ser usado para os dois tipos de tráfego. Se estiver enviando tráfego OTP ou 2FA, o ID de remetente deverá ser usado somente para essa finalidade. Os IDs de remetente promocionais estarão sujeitos à lista Não Perturbe (DND) da Arábia Saudita.	<ul style="list-style-type: none"> <li>Tamanho do ID de remetente promocional: 2 a 8 caracteres, anexado com “-AD”</li> <li>Tamanho do ID de remetente transaccional: 2 a 11 caracteres</li> <li>O ID de remetente deve representar a identidade da marca do remetente</li> <li>Incluir pelo menos uma letra</li> <li>Não use caracteres especiais ASCII (por exemplo, #, @)</li> <li>Você pode incluir letras maiúsculas e minúsculas e os números de 0 a 9</li> </ul>	<p>A compatibilidade com o registro de ID de remetente na Arábia Saudita é apenas para empresas internacionais. No momento, não oferecemos suporte a empresas locais da Arábia Saudita para registro de ID de remetente.</p> <ul style="list-style-type: none"> <li>O ID do remetente que você deseja registrar</li> <li>Região da AWS da qual o usuário chamará a API/ serviço</li> <li>Company name (Nome da empresa)</li> <li>Endereço da empresa (incluindo o cidade, estado/província e CEP)</li> <li>País da empresa</li> <li>Número de contato da empresa</li> <li>URL da empresa (link para a aplicação ou site da empresa)</li> </ul>

Nome do país	Tipo de mensagem	Restrições e requisitos de formato	Requisitos de registro
			<ul style="list-style-type: none"><li>• Volume mensal estimado</li><li>• Explicação do caso de uso e finalidade das mensagens</li><li>• Se não for óbvia, forneça uma explicação sobre a relação entre o nome da empresa e o ID de remetente</li><li>• Modelos de mensagens que você planeja enviar.</li><li>• Esse ID de remetente será usado para conteúdo transacional ou promocional?</li><li>• Confirmação de que mensagens não transacionais serão enviadas usando outro ID de remetente</li><li>• Se estiver enviando tráfego 2FA ou OTP, confirmação de que esse ID de remetente será</li></ul>

Nome do país	Tipo de mensagem	Restrições e requisitos de formato	Requisitos de registro
			usado SOMENTE para essa finalidade
Singapura (SG)	Somente mensagens transacionais	<ul style="list-style-type: none"><li>• Máximo de 11 caracteres</li><li>• Sem espaços</li><li>• Sem caracteres especiais</li></ul>	Consulte <a href="#">Requisitos para registro de ID de remetente de Singapura</a> .

Nome do país	Tipo de mensagem	Restrições e requisitos de formato	Requisitos de registro
Sri Lanka (LK)	Sem restrições ou requisitos especiais	<ul style="list-style-type: none"><li>• Alfanumérico</li><li>• Máximo de 11 caracteres</li><li>• Sem espaços</li><li>• Sem caracteres especiais</li></ul>	<ul style="list-style-type: none"><li>• O ID do remetente que você deseja registrar</li><li>• Região da AWS da qual o usuário chamará a API/ serviço</li><li>• Company name (Nome da empresa)</li><li>• Tipo de empresa (local/internacional)</li><li>• URL da empresa (link para a aplicação ou site da empresa)</li><li>• Explicação do caso de uso e finalidade das mensagens</li><li>• Modelos de mensagens que você planeja enviar</li><li>• Esse ID de remetente será usado para conteúdo transacional ou promocional?</li><li>• Confirmação de que mensagens não transacionais serão enviadas</li></ul>

Nome do país	Tipo de mensagem	Restrições e requisitos de formato	Requisitos de registro
			<p>usando outro ID de remetente</p> <ul style="list-style-type: none"><li>• Se estiver enviando tráfego 2FA ou OTP, confirmação de que esse ID de remetente será usado somente para essa finalidade</li></ul>

Nome do país	Tipo de mensagem	Restrições e requisitos de formato	Requisitos de registro
Tailândia (TH)	Sem restrições ou requisitos especiais	<ul style="list-style-type: none"><li>• Alfanumérico</li><li>• Máximo de 11 caracteres</li><li>• Sem espaços</li><li>• Sem caracteres especiais</li></ul>	<ul style="list-style-type: none"><li>• O ID do remetente que você deseja registrar</li><li>• Região da AWS da qual o usuário chamará a API/ serviço</li><li>• Company name (Nome da empresa)</li><li>• Endereço da empresa (incluindo o cidade, estado/província e CEP)</li><li>• País da empresa</li><li>• Número de contato da empresa</li><li>• URL da empresa (link para a aplicação ou site da empresa)</li><li>• Volume mensal estimado</li><li>• Explicação do caso de uso e finalidade das mensagens</li><li>• Se não for óbvia, forneça uma explicação sobre a relação entre o nome da empresa e o ID de remetente</li></ul>

Nome do país	Tipo de mensagem	Restrições e requisitos de formato	Requisitos de registro
			<ul style="list-style-type: none"><li>• Tipo de SMS enviado (transacional/promocional/OTP)</li><li>• Modelos de mensagens que você planeja enviar</li></ul>

Nome do país	Tipo de mensagem	Restrições e requisitos de formato	Requisitos de registro
Turquia (TR)	Sem restrições ou requisitos especiais	<ul style="list-style-type: none"><li>• Alfanumérico</li><li>• Máximo de 11 caracteres</li><li>• Sem espaços</li><li>• Sem caracteres especiais</li></ul>	<ul style="list-style-type: none"><li>• O ID do remetente que você deseja registrar</li><li>• Região da AWS da qual o usuário chamará a API/ serviço</li><li>• Company name (Nome da empresa)</li><li>• Endereço da empresa (incluindo o cidade, estado/província e CEP)</li><li>• URL da empresa (link para a aplicação ou site da empresa)</li><li>• Se sua empresa é local ou internacional na Turquia</li><li>• Volume mensal estimado</li><li>• Explicação do caso de uso e finalidade das mensagens</li><li>• Se não for óbvia, forneça uma explicação sobre a relação entre o nome da empresa e o ID de remetente</li></ul>

Nome do país	Tipo de mensagem	Restrições e requisitos de formato	Requisitos de registro
			<ul style="list-style-type: none"><li>• Tipo de SMS enviado (transacional/promocional/OTP)</li><li>• Modelos de mensagens que você planeja enviar</li></ul>

Nome do país	Tipo de mensagem	Restrições e requisitos de formato	Requisitos de registro
Ucrânia (UA)	Sem restrições ou requisitos especiais	<ul style="list-style-type: none"> <li>• Alfanumérico</li> <li>• Máximo de 11 caracteres</li> <li>• Sem espaços</li> <li>• Sem caracteres especiais</li> </ul>	<ul style="list-style-type: none"> <li>• O ID do remetente que você deseja registrar</li> <li>• Região da AWS da qual o usuário chamará a API/ serviço</li> <li>• Company name (Nome da empresa)</li> <li>• Endereço da empresa (incluindo o cidade, estado/província e CEP)</li> <li>• URL da empresa (link para a aplicação ou site da empresa)</li> <li>• Número de IVA</li> <li>• Empresa local ou internacional</li> <li>• Volume mensal estimado</li> <li>• Explicação do caso de uso e finalidade das mensagens</li> <li>• Se não for óbvia, forneça uma explicação sobre a relação entre o nome da empresa e o ID de remetente</li> </ul>

Nome do país	Tipo de mensagem	Restrições e requisitos de formato	Requisitos de registro
			<ul style="list-style-type: none"><li>• Tipo de SMS enviado (transacional/promocional/OTP)</li><li>• Modelos de mensagens que você planeja enviar</li></ul>

Nome do país	Tipo de mensagem	Restrições e requisitos de formato	Requisitos de registro
Emirados Árabes Unidos (AE)	Somente mensagens transacionais	<ul style="list-style-type: none"><li>• Alfanumérico</li><li>• IDs de remetente genéricos não são permitidos e devem identificar a marca</li><li>• Máximo de 11 caracteres</li><li>• Sem espaços</li><li>• Sem caracteres especiais</li></ul>	<ul style="list-style-type: none"><li>• O ID do remetente que você deseja registrar</li><li>• Região da AWS da qual o usuário chamará a API/ serviço</li><li>• Company name (Nome da empresa)</li><li>• Endereço da empresa (incluindo o cidade, estado/província e CEP)</li><li>• País da empresa</li><li>• Número de contato da empresa</li><li>• URL da empresa (link para a aplicação ou site da empresa)</li><li>• Volume mensal estimado</li><li>• Se não for óbvia, forneça uma explicação sobre a relação entre o nome da empresa e o ID de remetente</li><li>• Explicação do caso de uso e finalidade das mensagens</li></ul>

Nome do país	Tipo de mensagem	Restrições e requisitos de formato	Requisitos de registro
			<ul style="list-style-type: none"><li>• Número da licença empresarial oficial da empresa ou número de IVA</li><li>• Confirmação por escrito de que todo tráfego enviado por esse ID de remetente está descrito no caso de uso informado</li><li>• Modelos de mensagens que você planeja enviar</li></ul>

Nome do país	Tipo de mensagem	Restrições e requisitos de formato	Requisitos de registro
Vietnã (VN)	<p>Somente mensagens transacionais. Mensagens promocionais e de marketing não são permitidas. Exemplos de conteúdo proibido:</p> <ul style="list-style-type: none"> <li>• Conteúdo adulto</li> <li>• Serviços de caridade</li> <li>• Criptomoedas</li> <li>• Loteria</li> <li>• Jogos de azar/cassino para dispositivos móveis</li> <li>• Apostas esportivas</li> <li>• Votação</li> </ul>	<ul style="list-style-type: none"> <li>• Máximo de 11 caracteres</li> <li>• Sem espaços</li> <li>• Sem caracteres especiais</li> </ul>	<ul style="list-style-type: none"> <li>• O ID do remetente que você deseja registrar</li> <li>• Região da AWS da qual o usuário chamará a API/serviço</li> <li>• Volume mensal estimado</li> <li>• Se não for óbvia, forneça uma explicação sobre a relação entre o nome da empresa e o ID de remetente</li> <li>• Explicação do caso de uso e finalidade das mensagens</li> <li>• Confirmação por escrito de que todo tráfego enviado por esse ID de remetente está descrito no caso de uso informado</li> </ul>

### Restrições e requisitos de formato de assinatura

Para que a entrega seja eficaz na China, é necessário registrar uma assinatura de mensagem com seu modelo de conteúdo de mensagem. Essa assinatura de mensagem deve ser anexada ao conteúdo de cada SMS enviado. Se você não anexar a assinatura registrada ao corpo da mensagem SMS, o SMS poderá ser bloqueado ou filtrado.

- A assinatura deve ser anexada ao corpo do SMS entre colchetes
- O texto padrão deve estar entre colchetes
- Assinaturas em texto unicode devem estar entre colchetes lenticulares: U+3010 COLCHETE LENTICULAR ESQUERDO e U+3011 COLCHETE LENTICULAR DIREITO – Exemplo: [Aviso]
- Deve ter de 3 a 11 caracteres
- Caracteres chineses/ingleses são compatíveis

## Requisitos de identificação de remetente para a França

Este guia fornece as etapas e diretrizes necessárias para criar um ID de remetente dedicado exigido pelas operadoras móveis francesas para enviar mensagens de texto SMS para a França.

### Tópicos

- [Configurar um ID de remetente dedicado para a França](#)
- [Diretrizes de nomeação do ID do remetente](#)

## Configurar um ID de remetente dedicado para a França

É possível usar um dos métodos a seguir para configurar um ID de remetente dedicado. O Amazon SNS usará o ID do remetente em seu nome para mensagens SMS publicadas usando a API Publish.

- É possível usar o console do Amazon SNS para configurar o ID de remetente a ser usado para todas as mensagens SMS publicadas. Para saber mais, acesse [Configurando as preferências de mensagens SMS usando o AWS Management Console](#).
- Você pode usar a API Publish para definir a ID do remetente usando o atributo de mensagem `AWS.SNS.SMS.SenderID` ao solicitar que o Amazon SNS publique uma mensagem SMS. Para saber mais, acesse [Enviar uma mensagem \(console\)](#).

## Diretrizes de nomeação do ID do remetente

- O nome do ID do remetente deve ser alfanumérico com no máximo 11 caracteres.
- O nome do ID do remetente não deve conter caracteres especiais ou espaços.
- Recomendamos que você use o mesmo nome para o ID do remetente e o nome comercial da empresa que está enviando a mensagem de texto SMS.

## Requisitos para registro de ID de remetente da Índia

Por padrão, quando você envia mensagens para destinatários na Índia, o Amazon SNS usa conexões do International Long Distance Operator (ILDO – operador internacional de longa distância) para transmiti-las. Quando os destinatários veem uma mensagem enviada por uma conexão ILDO, ela aparece como tendo sido enviada de um ID numérico aleatório (a menos que você [compre um código simplificado dedicado](#)).

### Note

O preço para enviar mensagens usando rotas locais é exibido na página [Amazon SNS Worldwide SMS Pricing](#) (Preço global do SMS para o Amazon SNS). O preço para enviar mensagens usando conexões ILDO é mais caro que o preço para enviar mensagens por rotas locais.

Se você preferir usar um ID de remetente alfabético para as mensagens SMS, envie-as por rotas locais em vez de rotas ILDO. Para enviar mensagens usando rotas locais, primeiro é necessário registrar seu caso de uso e os modelos de mensagens na Telecom Regulatory Authority of India (TRAI – Autoridade reguladora de telecomunicações da Índia) por meio dos portais da Distributed Ledger Technology (DLT). Esses requisitos de registro foram criados para reduzir o número de mensagens não solicitadas recebidas por indianos e para proteger os consumidores contra mensagens potencialmente prejudiciais. Esse processo de registro é gerenciado pela Vodafone India por meio do serviço Vilpower.

### Tópicos

- [Etapa 1: inscrever-se na TRAI](#)
- [Etapa 2: solicitar um ID do remetente](#)
- [Etapa 3: enviar mensagens SMS](#)
- [Solução de problemas de mensagens SMS enviadas para destinatários na Índia](#)

### Etapa 1: inscrever-se na TRAI

Antes de enviar mensagens SMS para destinatários na Índia, é necessário inscrever sua organização na Telecom Regulatory Authority of India (TRAI). Prepare-se para fornecer as seguintes informações durante o processo de registro:

- Permanent Account Number (PAN – número da conta permanente) da organização.

- Tax Deduction Account Number (TAN – número da conta de dedução de imposto) da organização.
- Goods and Services Tax Identification Number (GSTIN – número de identificação do imposto de bens e serviços) da organização.
- Corporate Identity Number (CIN – número de identidade corporativa) da organização.
- Uma carta de autorização que concede autorização para você inscrever sua organização.

Veja a seguir uma lista de exemplos de alguns sites de registro da tecnologia de razão distribuído (DLT) que você pode usar para registrar sua organização na TRAI (taxas podem ser aplicadas). O processo de registro varia de acordo com o local. Entre em contato com as respectivas equipes de suporte para obter assistência.

- [BSNL DLT](#): registro gratuito.
- [Jio Trueconnect](#): cobra uma taxa pela conclusão do processo de registro.
- [Smart Enterprise Solutions](#): cobra uma taxa pela conclusão do processo de registro.
- [Vilpower](#): inclui um modelo que você pode baixar e modificar para ajustar às suas necessidades. O Vilpower cobra uma taxa para concluir o processo de registro.

Para registrar a sua organização na TRAI

A seguir são apresentados detalhes sobre como inscrever sua organização na TRAI usando o Vilpower.

1. Em um navegador da Web, acesse o site do Vilpower em <https://www.vilpower.in>.
2. Escolha Cadastre-se para criar outra conta. Durante o processo de registro, faça o seguinte:
  - Para o tipo de entidade a ser registrada, escolha As Enterprise (Como empresa).
  - Em Telemarketer Name (Nome do operador de telemarketing), use Infobip Private Limited - ALL. Quando solicitado, comece a digitar **Infobip** e, depois, escolha Infobip Private Limited – ALL na lista suspensa.
  - Em Inserir ID do operador de telemarketing, insira **110200001152**.
  - Quando for solicitado a inserir os IDs de cabeçalho, insira os IDs dos remetentes que deseja registrar.

**Note**

A Índia exige que os IDs de remetente tenham exatamente seis caracteres.

- Quando for solicitado a fornecer os modelos de conteúdo, insira o conteúdo da mensagem que pretende enviar para os destinatários. Inclua um modelo para cada mensagem que pretende enviar.

**Note**

Os sites do provedor de registro de DLT não são mantidos pela Amazon Web Services. As etapas no site estão sujeitas a alterações.

## Etapa 2: solicitar um ID do remetente

Para solicitar um ID do remetente na Índia, você precisa registrar uma solicitação AWS Support. Conclua as etapas em [Solicitar IDs do remetente](#). Em sua solicitação, forneça as seguintes informações obrigatórias:

- A região AWS da qual o remetente planeja enviar mensagens SMS.
- O nome da empresa usado durante o processo de registro da DLT.
- O Principal Entity ID (PEID – ID de entidade principal) que você recebeu após o registro de entidade da DLT bem-sucedido.
- Volumes mensais estimados.
- Uma explicação do seu caso de uso.
- Uma descrição do fluxo de inclusão do usuário final.
- Confirmação de que as inclusões de usuários finais são coletadas e registradas.

## Etapa 3: enviar mensagens SMS

Depois de [registrar sua organização na TRAI](#), é possível enviar mensagens SMS a destinatários na Índia.

1. Faça login no [console do Amazon SNS](#).

2. No menu do console, defina a seleção de região para uma [região que comporte mensagens SMS](#).
3. No painel de navegação, escolha Text messaging (SMS) [Mensagens de texto (SMS)].
4. Na página Mobile text messaging (SMS) (Mensagens de texto para dispositivos móveis (SMS)), selecione Publish text message (Publicar mensagem de texto). A janela Publish SMS message (Publicar mensagem SMS) é aberta.
5. Em Tipo de mensagem, escolha uma das seguintes opções:
  - Promotional (Promocional): mensagens não essenciais, como mensagens de marketing.

Ao usar IDs do remetente numéricos, escolha essa opção.

- Transactional (Transacional): mensagens urgentes que oferecem suporte para transações do cliente, como senhas únicas para autenticação multifator.

Ao usar IDs do remetente alfabéticos ou alfanuméricos, escolha essa opção.

Essa configuração no nível de mensagem substitui o tipo de mensagem padrão que você define na página Preferências de mensagens de texto.

Para obter informações sobre a definição de preços para mensagens promocionais e transacionais, consulte [Definição global de preço para SMS](#).

6. Em Número, insira o número de telefone para o qual você deseja enviar a mensagem.
7. Em Message (Mensagem), digite a mensagem a ser enviada.

Ao adicionar conteúdo a mensagens SMS, certifique-se de que ele corresponda exatamente ao conteúdo no modelo registrado na DLT. As operadoras bloqueiam mensagens SMS se o conteúdo da mensagem incluir retornos de caracteres adicionais, espaços, pontuação ou maiúsculas e minúsculas incompatíveis. As variáveis em um modelo podem ter 30 ou menos caracteres.

8. Na seção Identidades de origem, para a seção ID do remetente, insira um ID personalizado que contenha de 3 a 11 caracteres.

Os IDs do remetente podem ser numéricos para mensagens promocionais ou alfabéticos ou alfanuméricos para mensagens transacionais. O ID do remetente é exibido como o remetente da mensagem no dispositivo receptor.

Para IDs de remetente promocionais numéricos registrados na Índia, especifique o ID do remetente como o parâmetro do [Número de origem](#) na solicitação de envio de SMS.

9. Expanda a seção Country-specific attributes (Atributos específicos do país) e especifique os seguintes atributos obrigatórios para enviar mensagens SMS a destinatários na Índia:

- Entity ID (ID da entidade): o ID da entidade ou ID da Principal Entity (PE – Entidade principal) que você recebeu do órgão regulador para enviar mensagens SMS a destinatários na Índia.

É uma string personalizada de 1 a 50 caracteres fornecida pela TRAI que identifica, de maneira exclusiva, a entidade que você registrou junto à TRAI.

- Template ID (ID do modelo): o ID de modelo que você recebeu do órgão regulador para enviar mensagens SMS a destinatários na Índia.

É uma string personalizada de 1 a 50 caracteres fornecida pela TRAI que identifica, de maneira exclusiva, o modelo que você registrou junto à TRAI. O ID do modelo deve ser associado ao ID do remetente, que você especificou na etapa anterior, e ao conteúdo da mensagem.

10. Selecione Publish message (Publicar mensagem).

Para obter informações sobre como enviar mensagens SMS a destinatários em outros países, consulte [Publicar em um telefone celular](#).

Solução de problemas de mensagens SMS enviadas para destinatários na Índia

Veja alguns motivos pelos quais as operadoras podem bloquear mensagens SMS:

- No template was found that matched the content sent. (Nenhum modelo foi encontrado que correspondesse ao conteúdo enviado.)

Conteúdo enviado: **<#> 12345 is your OTP to verify mobile number. Your OTP is valid for 15 minutes -- ABC Pvt. Ltd.**

Modelo correspondente: nenhum

Problema: não há modelos da DLT que incluam **<#>** ou **{#var#}** no início do modelo registrado junto à DLT.

- The value of a variable exceeds 30 characters. (O valor de uma variável excede 30 caracteres.)

Conteúdo enviado: **12345 is your OTP code for ABC (ABC Company - India Private Limited) - (ABC 123456789). Share with your agent only. - ABC Pvt. Ltd.**

Modelo correspondente: **{#var#} is your OTP code for {#var#} ({#var#}) - ({#var#} {#var#}). Share with your agent only. - ABC Pvt. Ltd.**

Problema: o valor de “ABC Company - India Private Limited” no conteúdo enviado excede o limite de 30 caracteres de uma única {#var#}.

- The message sentence case does not match the sentence case in the template. (O uso de maiúsculas e minúsculas da frase da mensagem não corresponde ao uso empregado no modelo.)

Conteúdo enviado: **12345 is your OTP code for abc (abc company - india private limited) - (abc 123456789). share with your agent only. - abc pvt. ltd.**

Modelo correspondente: **{#var#} is your OTP code for {#var#} ({#var#}) - ({#var#} {#var#}). Share with your agent only. - ABC PVT. LTD.**

Problema: o nome da empresa anexado ao modelo correspondente de DLT está com letras maiúsculas enquanto o conteúdo enviado mudou partes do nome para minúsculas — “ABC Pvt. Ltd.” vs. “ABC PVT. LTD.”

## Requisitos para registro de ID de remetente de Singapura

Os clientes do Amazon SNS podem enviar tráfego de SMS em Singapura usando um ID do remetente registrado por meio do Registro de ID de remetentes de SMS de Singapura (SSIR). O SSIR foi lançado em março de 2022 pelo Singapore Network Information Centre (SGNIC), que pertence à Info-communications Media Development Authority (IMDA) de Singapura, e permite que as organizações registrem o ID de remetente ao enviar SMS para telefones celulares em Singapura.

Para usar um ID de remetente registrado em Singapura, você deve obter um Unique Entity Number (UEN), depois enviar uma solicitação à Amazon para que sua conta seja incluída na lista de permissões para uso do seu ID de remetente e, por fim, concluir o processo de registro por meio do SSIR.

Se você não registrar seu ID até 30/1/2023, todas as mensagens enviadas usando um ID de remetente terão o ID alterado para LIKELY-SCAM de acordo com as regras do órgão regulador.

Após essa data, os reguladores continuarão filtrando ou bloqueando o tráfego não registrado a seu critério.

### Important

Se você estiver solicitando o ID de remetente nas [regiões do Amazon Pinpoint](#), use o [console do Amazon Pinpoint](#) para registrar o ID de remetente. Para concluir o processo de registro manualmente para regiões que não sejam do Amazon Pinpoint, use o [registro de ID de remetente de Singapura](#).

Para garantir que você ainda possa enviar mensagens em Singapura, seu registro deve ser concluído até 30/01/2023.

É muito importante que você conclua as etapas de registro na ordem a seguir. A execução dessas etapas fora de ordem pode resultar no bloqueio do seu ID de remetente pelo serviço ou impedir que seu ID de remetente seja preservado no dispositivo móvel.

Etapa 1. [Como se registrar para obter um Unique Entity Number \(UEN\) de Singapura](#)

Etapa 2. Se você estiver solicitando o ID de remetente nas [regiões do Amazon Pinpoint](#), use as instruções para [registro de ID de remetente do Amazon Pinpoint](#) a fim de registrar o ID de remetente.

- Para registrar um ID de remetente em que a conta não está em uma [região do Amazon Pinpoint](#), use as instruções para [registro de ID de remetente de Singapura](#) a fim de registrar manualmente o ID de remetente.
- Ao enviar mensagens de texto SMS em nome de outra empresa, é necessária uma carta de autorização (LOA) da empresa.
- Não espere por uma aprovação ou mudança de status depois de enviar seu registro de ID de remetente da AWS. Vá imediatamente para a Etapa 3.

Etapa 3. [Como registrar um ID de remetente no Singapore Network Information Centre \(SGNIC\)](#)

## Tópicos

- [Como se registrar para obter um Unique Entity Number \(UEN\) de Singapura](#)
- [Como registrar seu ID de remetente em Singapura no Amazon Pinpoint](#)
- [Processo manual para concluir o registro de ID de remetente de Singapura](#)
- [Como registrar um ID de remetente no Singapore Network Information Centre \(SGNIC\)](#)

- [Status de registro de ID de remetente de Singapura](#)
- [Editar um registro de ID de remetente de Singapura](#)
- [Excluir um registro de ID de remetente de Singapura](#)
- [Problemas de registro em Singapura](#)
- [Perguntas frequentes sobre o registro de ID de remetente de Singapura](#)

Como se registrar para obter um Unique Entity Number (UEN) de Singapura

Para iniciar um registro no SSIR, primeiro você deve obter um Unique Entity Number (UEN) de Singapura. Um UEN é um número de entidade exclusivo que você recebe ao registrar sua empresa na Account and Corporate Registry Authority (ACRA). Para obter mais informações, consulte [Quem deve se registrar na ACRA?](#) O tempo de processamento pode variar dependendo da facilidade com que a ACRA consegue validar sua solicitação.

Como registrar seu ID de remetente em Singapura no Amazon Pinpoint

Depois de registrar seu Unique Entity Number (UEN) de Singapura, você pode concluir o processo de registro do ID de remetente no console do Amazon Pinpoint (somente para as [regiões do Amazon Pinpoint](#)). Ao registrar seu ID de remetente, verifique se as informações estão completas e corretas, ou seu registro poderá ser rejeitado.

 Important

As informações que você enviar por meio do console do Amazon Pinpoint serão repassadas às nossas operadoras parceiras para concluir o registro.

Como registrar um ID de remetente de Singapura

Use estas etapas para registrar um ID de remetente quando a conta está em uma [região do Amazon Pinpoint](#). Se sua conta não estiver em uma região do Amazon Pinpoint, consulte [Processo manual para concluir o registro de ID de remetente de Singapura](#).

1. Faça login no Console de Gerenciamento da AWS e abra o console do Amazon Pinpoint em <https://console.aws.amazon.com/pinpoint/>.
2. No painel de navegação, em SMS and voice (SMS e voz), selecione Phone numbers (Números de telefone).

3. Na guia Sender ID registrations (Registros de ID de remetente), escolha Create registration (Criar registro).
4. Escolha Singapore (Singapura) como seu país de destino.
5. Na seção Company Information (Informações da empresa), insira o seguinte:
  - Em Company name (Nome da empresa), insira o nome da sua empresa exatamente como aparece no seu registro de UEN.
  - Em Tax ID (ID fiscal), insira o número UEN que você recebeu da ACRA.
  - Em Company Website (Site da empresa), insira o URL do site de sua empresa.
  - Em Address 1 (Endereço 1), insira o endereço da sede corporativa.
  - Se necessário, em Address 2 - optional (Endereço 2 - opcional), insira o número do conjunto da sede corporativa.
  - Em City (Cidade), insira a cidade da sede corporativa.
  - Em State (Estado), insira o estado da sede corporativa.
  - Em Zip Code (CEP), insira o CEP da sede corporativa.
  - Em Country (País), insira o código de país ISO de dois dígitos.
6. Na seção Contact Information (Informações para contato), insira as seguintes informações:
  - Em First Name (Nome), insira o nome da pessoa que será o ponto de contato da empresa.
  - Em Last Name (Sobrenome), insira o nome da pessoa que será o ponto de contato da empresa.
  - Em Support Email (E-mail de suporte), insira o endereço de e-mail da pessoa que será o ponto de contato da empresa.
  - Em Support Phone Number (Telefone do suporte), insira o número de telefone da pessoa que será o ponto de contato da empresa.
7. Em Sender ID Information (Informações de ID de remetente), insira o seguinte:
  - Em Sender ID (ID de remetente), insira o ID de remetente que você deseja exibir para suas mensagens.
  - Em Registering on behalf of another brand/entity? (Registro em nome de outra marca/entidade?), selecione True em caso positivo. Se você não for o usuário final que envia as mensagens, será considerado um "Representante" da outra marca/entidade.
  - Em Letter of authorization image – opcional (Imagem da carta de autorização – opcional), se você marcou a caixa Registering on behalf of another brand/entity? (Registro em nome de outra marca/entidade?), faça upload de uma imagem da Carta de Autorização (LOA) completa. O tipo

de arquivo compatível é PNG e o tamanho máximo é 400 KB. Você pode [fazer download](#) de um modelo de LOA para sua conveniência.

- Em Sender ID connection – optional (Conexão do ID de remetente – opcional), você pode adicionar mais detalhes sobre a conexão entre o ID de remetente solicitado e o nome da empresa.

#### 8. Em Messaging Use Case (Caso de uso de mensagens), faça o seguinte:

- Em Monthly SMS Volume (Volume mensal de SMS), selecione o número de mensagens SMS que serão enviadas a cada mês.
- Em Use Case Category (Categoria do caso de uso), selecione um dos seguintes tipos de caso de uso para o número:
  - Two-factor authentication (Autenticação de dois fatores): use para enviar códigos de autenticação de dois fatores.
  - One-time passwords (Senhas de uso único): use para enviar a um usuário uma senha de uso único.
  - Notifications (Notificações): use se você só pretende enviar notificações importantes aos seus usuários.
  - Polling and surveys (Sondagens e pesquisas): use para pesquisar as preferências dos usuários.
  - Info on demand (Informações sob demanda): essa opção serve para enviar mensagens aos usuários depois que eles enviam uma solicitação.
  - Promotions and Marketing (Promoções e marketing): use se você só pretende enviar mensagens de marketing para seus usuários.
  - Other (Outros): use se seu caso de uso não se enquadrar em nenhuma outra categoria. Preencha os Use Case Details (Detalhes de caso de uso) para essa opção.
- Preencha Use Case Details – optional (Detalhes de caso de uso – opcional) para fornecer contexto adicional sobre a Use Case Category (Categoria de caso de uso) selecionada.

#### 9. Na seção Messaging Samples (Exemplos de mensagens), faça o seguinte:

- Em Message Sample 1 (Exemplo de mensagem 1), insira um exemplo do corpo de uma mensagem SMS que será enviada aos usuários finais.
- Se necessário, em Message Sample 2 – optional (Exemplo de mensagem 2 – opcional) e Message Sample 3 – optional (Exemplo de mensagem 3 – opcional), insira mais exemplos de corpos de mensagens SMS que serão enviadas.

- Cada caixa de Message Sample (Exemplo de mensagem) tem um limite máximo de 306 caracteres.

10. Quando você terminar, escolha Submit registration (Enviar registro).

**⚠ Important**

Você pode verificar o status do seu registro seguindo as instruções em [Status de registro de ID de remetente de Singapura](#).

Não espere por uma aprovação ou mudança de status depois de enviar seu registro de ID de remetente. Vá imediatamente para [Como registrar um ID de remetente no Singapore Network Information Centre \(SGNIC\)](#).

Processo manual para concluir o registro de ID de remetente de Singapura

Use estas etapas para registrar um ID de remetente quando a conta não está em uma [região do Amazon Pinpoint](#). Se sua conta estiver em uma região do Amazon Pinpoint, consulte [Como registrar seu ID de remetente em Singapura no Amazon Pinpoint](#).

1. Faça download do arquivo [Singapore\\_Sender\\_ID\\_Registration\\_LOA\\_Template.zip](#) e preencha as informações necessárias.
2. Crie um caso em [AWS Support](#).
3. Na guia Open support cases (Solicitações de suporte em aberto), escolha Create case (Criar solicitação).
4. Escolha Looking for service limit increases (Em busca de aumentos no limite de serviço); para o tipo de limite, escolha SNS Text Messaging (Mensagens de texto do SNS).
5. Em Resource Type (Tipo de recurso), escolha Sender ID Registration (Registro de ID de remetente).
6. Anexe o documento LOA e envie a solicitação.

Como registrar um ID de remetente no Singapore Network Information Centre (SGNIC)

**⚠ Warning**

A execução dessas etapas fora de ordem pode resultar no bloqueio do seu ID de remetente pelo serviço ou impedir que seu ID de remetente seja preservado no dispositivo móvel.

1. Primeiro você deve registrar seu ID de remetente de Singapura (SG) para sua conta na AWS ([console do Amazon Pinpoint](#), ou [registro manual](#) para regiões que não sejam do Amazon Pinpoint). Quando essa etapa estiver concluída, você poderá avançar para a etapa seguinte.
2. Entre em contato com o SGNIC para registrar seu ID de remetente usando o processo em [SGNIC SMS Sender ID Registry](#).
  - Ao concluir o processo, liste todos os seguintes itens como seus agregadores participantes:
    - AMCS SG Private Limited (Amazon Media Communications Services)
    - Nexmo PTE LTD
    - Sinch Singapore PTE LTD
    - Telesign Singapore PTE LTD
    - Twilio Singapore PTD LTD

**Note**

É necessário enviar um registro de ID de remetente de todas as contas da AWS específicas que deverão usar o ID do remetente.

### Status de registro de ID de remetente de Singapura

Ao registrar seu ID de remetente de Singapura no Amazon SNS, seu registro terá um destes cinco status:

- Created (Criado): seu registro foi criado, mas não foi enviado.
- Submitted (Enviado): seu registro foi enviado e está sendo validado.
- Reviewing (Em análise): seu registro foi aceito e está sendo analisado. Pode levar de 1 a 3 semanas e, em alguns casos, pode levar ainda mais tempo para que a revisão seja concluída.
- Complete (Concluído): seu registro foi aprovado e você pode começar a usar o ID de remetente.
- Require Updates (Requer atualizações): você deve corrigir seu registro e reenviá-lo. Consulte [Editar um registro de ID de remetente de Singapura](#) para obter mais informações. Os campos que exigem atualizações exibem um ícone de aviso e uma breve descrição do problema.

Para todas as regiões diferentes das [regiões do Amazon Pinpoint](#), o [AWS Support](#) enviará uma confirmação por e-mail no momento do registro ou criará um caso no [AWS Support](#).

- Na guia Open support cases (Solicitações de suporte em aberto), escolha Create case (Criar solicitação).
- Escolha Service limit increase (Aumento de limite do serviço).
- Em Resource Type (Tipo de recurso), escolha Sender ID Registration (Registro de ID de remetente) e, para limite, escolha General Inquiry (Consulta geral).

### Verificar o status de seu registro

1. Faça login no Console de Gerenciamento da AWS e abra o console do Amazon Pinpoint em <https://console.aws.amazon.com/pinpoint/>.
2. No painel de navegação, em SMS and voice (SMS e voz), selecione Phone numbers (Números de telefone).
3. Na guia Sender ID registrations (Registros de ID de remetente), escolha o SenderID (ID de remetente).
4. É possível visualizar o status do registro de cada SenderID (ID de remetente).

### Editar um registro de ID de remetente de Singapura

Depois de enviar seu registro no Amazon Pinpoint, se houver algum problema com ele, o status do registro será definido como Requires Updates (Requer atualizações). Nesse estado, o formulário de registro é editável. Os campos que exigem atualizações terão um ícone de aviso e uma breve descrição do problema.

### Como editar um ID de remetente

1. Faça login no console do Amazon Pinpoint em <https://console.aws.amazon.com/pinpoint/>.
2. No painel de navegação, em SMS and voice (SMS e voz), selecione Phone numbers (Números de telefone).
3. Na guia SenderID Registration (Registro de ID de remetente), escolha o número que você deseja editar e selecione o Registration ID (ID de registro).
4. Escolha Update registration (Atualizar registro) para editar o formulário e corrigir os campos que têm um ícone de aviso.
5. Se você estiver registrando em nome de outra marca/entidade, precisará fazer upload novamente dos arquivos enviados anteriormente para Letter of authorization image – optional (Imagem da carta de autorização – opcional).

6.  **Important**  
Verifique novamente todos os campos para garantir que estejam corretos.

7. Escolha Submit registration (Enviar registro) para reenviar ao concluir.

## Excluir um registro de ID de remetente de Singapura

Se você não quiser mais continuar com seu registro de ID de remetente de Singapura, poderá excluir o registro. Um registro só poderá ser excluído se o status for Created (Criado) ou Requires Updates (Atualizações necessárias).

### Como excluir um registro

1. Faça login no console do Amazon Pinpoint em <https://console.aws.amazon.com/pinpoint/>.
2. No painel de navegação, em SMS and voice (SMS e voz), selecione Phone numbers (Números de telefone).
3. Na guia Sender ID (ID de remetente), escolha o ID de registro que deseja excluir, depois selecione Delete Registration (Excluir registro).

## Problemas de registro em Singapura

Se o ID de remetente de Singapura não for aceito pelo Amazon Pinpoint, você receberá uma mensagem explicando o motivo da rejeição. Se você tiver dúvidas sobre essas rejeições que não estejam respondidas em nossas [Práticas recomendadas](#), envie uma solicitação às nossas equipes de suporte.

### Como enviar uma solicitação de informações sobre um ID de remetente de Singapura rejeitado

1. Faça login no console do Amazon Pinpoint em <https://console.aws.amazon.com/pinpoint/>.
2. No menu Support (Suporte), escolha Support Center (Central de Suporte).
3. Na página Support (Suporte), escolha Create Case (Abrir caso).
4. Em Case Type (Tipo de caso), escolha Service limit increase (Aumento do limite de serviço).
5. Em Limit type (Tipo de limite), selecione Pinpoint SMS (Identificar SMS).
6. Na seção Requests (Solicitações), faça o seguinte:
  - Em Resource Type (Tipo de recurso), escolha Sender ID Registration (Registro de ID de remetente).

- Em Limit (Limite), escolha Registration Rejection Query (Consulta de rejeição de registro).
7. Em Use case description (Descrição do caso de uso), insira o ID de remetente de Singapura rejeitado e o motivo fornecido para rejeição.
  8. Em Contact options (Opções de contato), em Preferred contact language (Idioma de contato preferido), escolha o idioma que você prefere usar ao se comunicar com a equipe de suporte da AWS.
  9. Em Contact method (Método de contato), selecione seu método preferido de comunicação com a equipe de suporte da AWS.
  10. Selecione Submit (Enviar).

A equipe do AWS Support fornecerá informações sobre os motivos pelos quais seu registro de ID de remetente foi rejeitado em seu caso do AWS Support.

Perguntas frequentes sobre o registro de ID de remetente de Singapura

Perguntas frequentes sobre o processo de registro de ID de remetente de Singapura no Amazon Pinpoint.

Eu já tenho um ID de remetente de Singapura?

Como verificar se você tem um ID de remetente de Singapura

1. Faça login no console do Amazon Pinpoint em <https://console.aws.amazon.com/pinpoint/>.
2. No painel de navegação, em SMS and voice (SMS e voz), selecione Phone numbers (Números de telefone).
3. Na guia SenderID Registration (Registro de ID de remetente), escolha o ID de remetente que você deseja exibir e selecione o Registration ID (ID de registro).

Quanto tempo leva para registrar?

Embora uma revisão típica leve de 1 a 3 semanas, a verificação de suas informações por agências governamentais pode levar até 5 semanas ou mais, em alguns casos.

O que é um Unique Entity Number (UEN) e como consigo um?

Um UEN é um ID comercial de Singapura emitido pela Accounting and Corporate Regulatory Authority (ACRA). Empresas e negócios locais em Singapura podem obter um UEN inscrevendo-se

pela ACRA. Depois de passar pelo procedimento de registro e incorporação padrão, ele será emitido. Você pode solicitar um UEN com a ACRA via [Bizfile](#).

Preciso me registrar para obter um ID de remetente de Singapura?

Sim. Se você não registrar seu ID de remetente de Singapura até 30/01/2023, todas as mensagens enviadas usando um ID de remetente terão o ID alterado para LIKELY-SCAM.

Como faço para registrar meu ID de remetente de Singapura no Amazon Pinpoint?

Siga as instruções em “Como registrar seu ID de remetente em Singapura no Amazon Pinpoint” para registrar um ID de remetente.

Qual é o status de registro do meu ID de remetente de Singapura e o que ele significa?

Siga as instruções em Status de registro de ID de remetente de Singapura para conferir seu registro e status.

Quais informações preciso fornecer?

Você precisará fornecer o endereço de sua empresa, um contato comercial e um caso de uso. Você pode encontrar as informações necessárias em “Como registrar seu ID de remetente em Singapura no Amazon Pinpoint”.

E se meu registro de ID de remetente de Singapura for rejeitado?

Se o seu registro for rejeitado, o status será alterado para Requires Updates (Requer atualizações) e você poderá fazer atualizações seguindo as instruções em Editar um registro de ID de remetente de Singapura.

De quais permissões eu preciso?

O usuário/perfil do IAM usado para acessar o console do Amazon Pinpoint precisará ter a permissão `“sms-voice:*”`.

## Números de origem

Um número de origem é uma cadeia numérica que identifica o número de telefone do remetente da mensagem SMS. Quando você envia uma mensagem SMS usando um número de origem, o dispositivo do destinatário mostra o número de origem como o número de telefone do remetente. É possível especificar diferentes números de origem por caso de uso.

 Tip

Para visualizar uma lista de todos os números de origem existentes em sua conta da AWS, no painel de navegação do [console do Amazon SNS](#), escolha Origination numbers (Números de origem).

O suporte para números de origem não está disponível em países onde as leis locais exigem o uso de [IDs remetente](#) em vez de números de origem.

## Tópicos

- [10DLC](#)
- [Números de chamada gratuita](#)
- [Códigos curtos](#)
- [Códigos longos de pessoa a pessoa \(P2P\)](#)
- [Comparação entre números de produtos dos EUA](#)

## 10DLC

As operadoras dos EUA não aceitam o uso de mensagens SMS de aplicação para pessoa (A2P) em códigos longos locais e não registrados. Para mensagens SMS A2P de alto volume, as operadoras dos EUA oferecem um novo tipo de código longo chamado 10-Digit Long Code (10DLC: código longo de 10 dígitos).

 Important

A partir de 26 de janeiro de 2023, os fornecedores de SMS do Amazon SNS apresentaram novos processos de revisão manual em campanhas de 10DLC para resolver questões de spam de SMS levantadas por operadoras dos EUA. É possível usar códigos curtos e números gratuitos como alternativas ao 10DLC para enviar SMS nos Estados Unidos. Atualmente, nossos fornecedores de SMS não forneceram metas de nível de serviço sobre quanto tempo levarão as análises da campanha de 10DLC. As análises são acionadas quando um número é associado a uma campanha de 10DLC. As análises estão demorando mais do que o tempo estimado de 14 dias que o Amazon SNS comunicou anteriormente. O Amazon SNS trabalha diariamente com fornecedores de SMS para garantir que:

- Os fornecedores conclua todas as análises pendentes da campanha de 10DLC o mais rapidamente possível
- Os fornecedores priorizem as solicitações da AWS em suas listas de pendências

Você pode verificar o status das campanhas de 10DLC seguindo as instruções em [Campanhas de 10DLC](#). Se for necessário obter mais informações para aprovar uma campanha de 10DLC, a equipe de suporte da AWS informará você.

Talvez seja mais rápido você registrar um número gratuito nos EUA do que obter números de 10DLC. Para obter mais informações sobre os números gratuitos dos EUA e o processo de registro, consulte [Requisitos e processo de registro de número de chamada gratuita](#).

## O que é o 10DLC?

10DLC é um tipo de código longo registrado junto às operadoras para comportar mensagens SMS A2P de alto volume usando o formato de número de telefone de 10 dígitos. O Amazon SNS não oferece mais códigos longos locais como produto SMS. Em vez disso, oferece o 10DLC. Se usar apenas códigos curtos e números gratuitos, o 10DLC não afetará você.

O 10DLC é um número de telefone de 10 dígitos usado apenas nos Estados Unidos. As mensagens enviadas de um 10DLC para destinatários mostram um número de 10 dígitos como o remetente. Ao contrário dos números de ligação gratuita, o 10DLC aceita tanto mensagens transacionais quanto promocionais e pode incluir qualquer código de área dos EUA.

Se você já tem códigos longos locais, pode solicitar que eles sejam habilitados para 10DLC. Para fazer isso, conclua o processo de registro de 10DLC e envie um tíquete de suporte. Se houver um problema com a habilitação do código longo para 10DLC, uma notificação será emitida e você receberá a instrução de solicitar um novo 10DLC por meio do console do Amazon Pinpoint (não do Amazon SNS). Para obter informações sobre como abrir um tíquete de suporte para converter um código longo, consulte [Associar um código longo a uma campanha 10DLC](#).

Para usar um número 10DLC, o primeiro passo é registrar sua empresa e criar uma campanha 10DLC usando o console do Amazon Pinpoint (não do Amazon SNS). A AWS compartilha essas informações com o The Campaign Registry, uma empresa externa que analisa as informações para aprovar ou rejeitar seu registro. Em alguns casos, o registro é realizado imediatamente. Por exemplo, se você já se inscreveu antes no The Campaign Registry, talvez eles já tenham suas informações. Porém, algumas campanhas podem levar uma semana ou mais para serem aprovadas. Após a

aprovação de sua empresa e da campanha 10DLC, é possível adquirir um número 10DLC e associá-lo à sua campanha. A aprovação de uma solicitação de 10DLC também pode levar até uma semana. Embora você possa associar vários 10DLCs a uma única campanha, não é possível usar um mesmo 10DLC em várias campanhas. Cada campanha criada precisa de um 10DLC exclusivo.

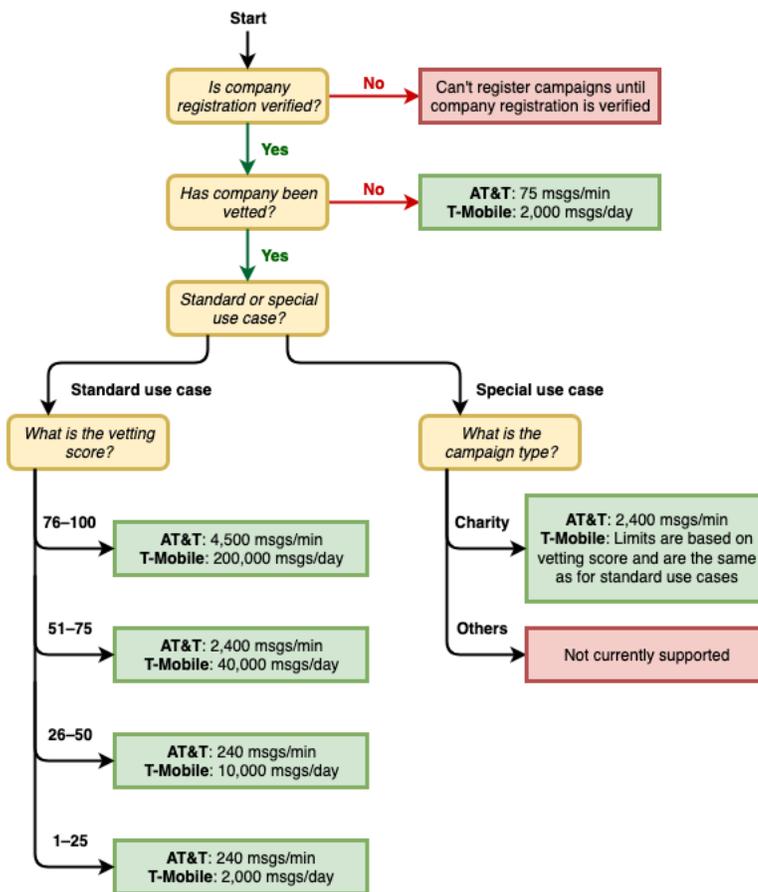
## Recursos do 10DLC

Os recursos dos números de telefone 10DLC dependem da operadora móvel que seus destinatários usam. A AT&T oferece um limite baseado em quantas partes de mensagens podem ser enviadas a cada minuto para cada campanha. A T-Mobile oferece um limite diário de mensagens que podem ser enviadas para cada campanha, sem limite para o número de partes de mensagens que podem ser enviadas por minuto. A Verizon não publicou limites de throughput, mas usa um sistema de filtragem para 10DLC projetado para remover spam, mensagens não solicitadas e conteúdo abusivo, com menor ênfase no throughput real da mensagem.

Novas campanhas 10DLC associadas a empresas não avaliadas podem enviar 75 partes de mensagens por minuto para destinatários que usam a AT&T e 2 mil mensagens por dia para destinatários que usam a T-Mobile. O limite da empresa é compartilhado em todas as suas campanhas 10DLC. Por exemplo, se você tiver inscrito uma empresa e duas campanhas, a alocação diária de 2 mil mensagens para clientes da T-Mobile será compartilhada entre essas campanhas. Da mesma forma, se você inscrever a mesma empresa em mais de uma conta da AWS, a alocação diária será compartilhada entre essas contas.

Se suas necessidades de throughput superarem esses limites, você poderá solicitar que o registro de sua empresa seja avaliado. Quando você verifica o registro de sua empresa, um provedor de verificação terceirizado analisa os respectivos detalhes. Esse provedor fornece uma pontuação de avaliação, que determina os recursos das campanhas 10DLC. Há uma cobrança única para o serviço de avaliação. Para obter mais informações, consulte [Análise de seu registro 10DLC no Amazon SNS](#).

Seu throughput real variará de acordo com diversos fatores, como se a empresa foi avaliada ou não, os tipos de campanha e a pontuação de avaliação. O fluxograma a seguir mostra o throughput para várias situações.



O throughput para 10DLC é determinado pelas operadoras móveis dos EUA em conjunto com o Campaign Registry. Nem o Amazon SNS nem nenhum outro serviço de envio de SMS podem aumentar o throughput de 10DLC para um valor além dessas taxas. Se você precisar de taxas altas de throughput e de capacidade de entrega em todas as operadoras dos EUA, é recomendável usar um código curto. Para obter mais informações sobre como obter um código curto, consulte [Solicitar códigos curtos dedicados para o sistema de mensagens por SMS com o Amazon SNS](#).

## Conceitos básicos do 10DLC

Use o console do [Amazon Pinpoint](#) (não do Amazon SNS) para solicitar o 10DLC. Siga as etapas abaixo para configurar o 10DLC e usá-lo em suas respectivas campanhas.

### 1. Inscreva sua empresa.

Para solicitar um 10DLC, primeiro sua empresa precisa ser inscrita no The Campaign Registry. Para obter informações, consulte [Registrar uma empresa](#). Normalmente, o registro é instantâneo, a menos que o The Campaign Registry exija mais informações. Para inscrever sua empresa, há

uma taxa de inscrição única exibida na página de registro. Essa taxa única é paga separadamente das cobranças mensais da campanha e do 10DLC.

 Note

As mensagens SMS do Amazon SNS estão disponíveis nas regiões em que o Amazon Pinpoint não é oferecido no momento. Há dois casos diferentes:

- a. Se você estiver usando uma conta comercial na nuvem, precisará abrir o console do [Amazon Pinpoint](#) na região Leste dos EUA (Norte da Virgínia) para registrar a campanha e a empresa 10DLC. Não solicite um número de 10DLC.
- b. Use o console do [AWS Service Quotas](#) para criar um caso de aumento de limite de serviço ao solicitar o número de 10DLC para essa região. Para obter informações sobre as regiões em que o Amazon Pinpoint está disponível, consulte [Endpoints e cotas do Amazon Pinpoint](#) na Referência geral da AWS.
- c. Se você estiver usando uma conta de AWS GovCloud (US), abra o console do [Amazon Pinpoint](#) na região Oeste dos EUA para registrar a campanha e a empresa 10DLC. Não solicite um número de 10DLC. Em vez disso, use o console do Service Quotas da AWS para criar um caso de aumento de limite de serviço ao solicitar o número 10DLC para essa região. Para obter informações sobre as regiões em que o Amazon Pinpoint está disponível, consulte [Endpoints e cotas do Amazon Pinpoint](#) na Referência geral da AWS.

## 2. (Opcional, mas recomendado) Solicite uma avaliação

Se a inscrição de sua empresa for bem-sucedida, você poderá começar a criar campanhas 10DLC de baixo volume e uso misto. Essas campanhas podem enviar 75 mensagens por minuto para destinatários que usam a AT&T, e a empresa inscrita pode enviar 2 mil mensagens por dia para destinatários que usam a T-Mobile. Se seu caso de uso exigir um throughput superior a esses valores, você poderá solicitar a avaliação da inscrição de sua empresa. A avaliação da inscrição pode aumentar o throughput para suas empresas e campanhas, mas isso não é garantido. Para obter mais informações sobre avaliação, consulte [Análise de seu registro 10DLC no Amazon SNS](#).

## 3. Inscreva sua campanha.

Depois que você inscrever sua empresa, crie uma campanha 10DLC e a associe a uma de suas empresas inscritas. Essa campanha é enviada ao The Campaign Registry para aprovação. Na maioria dos casos, a aprovação da campanha 10DLC é instantânea, a menos que o The

Campaign Registry exija mais informações. Para obter mais informações, consulte [Registrar uma campanha 10DLC](#).

#### 4. Solicite seu número 10DLC.

Após a aprovação de sua campanha 10DLC, é possível solicitar um número 10DLC e associá-lo à campanha aprovada. A campanha 10DLC pode usar apenas o número aprovado para ela. Consulte [Solicitar números 10DLC, números de chamada gratuita e códigos longos P2P para mensagens SMS com o Amazon SNS](#).

#### Taxa de inscrição e taxas mensais do 10DLC

Há uma taxa de inscrição e taxas mensais associadas ao uso do 10DLC, como a inscrição de sua empresa e da campanha 10DLC. Essas taxas são distintas de todas as outras taxas mensais cobradas pela AWS. Para obter mais informações, consulte a página [Preço mundial de SMS do Amazon SNS](#).

#### Registrar uma empresa

Antes que você possa solicitar um 10DLC, sua empresa deve estar registrada no The Campaign Registry.

#### Note

As mensagens SMS do Amazon SNS estão disponíveis nas regiões em que o Amazon Pinpoint não tem suporte no momento. Nesses casos, abra o console do Amazon Pinpoint na região Leste dos EUA (Norte da Virgínia) para registrar a empresa e a campanha de seu 10DLC, mas não solicite um número 10DLC. Em vez disso, use o [console do Service Quotas da AWS](#) para criar um caso de aumento de limite de serviço ao solicitar o número 10DLC para essa região. Para obter informações sobre as regiões em que o Amazon Pinpoint está disponível, consulte [Endpoints e cotas do Amazon Pinpoint](#) na Referência geral da AWS.

#### Status de registro de empresa 10DLC

Quando você inscreve sua empresa ou marca, um dos dois status é retornado: Unverified (Não verificado) ou Verified (Verificado). Se o status do registro de sua empresa for Unverified (Não verificado), isso significa que houve um problema com seu registro. Por exemplo, o nome da empresa registrada que você forneceu pode não corresponder exatamente ao nome registrado

da empresa associada ao ID fiscal que você forneceu. Se você encontrar algum problema com os detalhes de registro de sua empresa, poderá corrigi-los. Para obter mais informações sobre como modificar detalhes de registro da empresa, consulte [Editar ou excluir uma empresa registrada](#).

Se o status de registro de sua empresa for Verified (Verificado), os detalhes de registro fornecidos foram precisos e você poderá começar a criar campanhas 10DLC.

### Inscrição de sua empresa ou marca

Você precisa inscrever sua empresa apenas uma vez. Depois de inscrita, você pode editar sua empresa e informações de contato. Para excluir uma empresa inscrita, abra um caso no [AWS Support](#). Para obter mais informações sobre como editar ou excluir detalhes da empresa, consulte [Editar ou excluir uma empresa registrada](#).

Para inscrever uma empresa

1. Faça login no AWS Management Console e abra o console do Amazon Pinpoint em <https://console.aws.amazon.com/pinpoint/>.
2. No painel de navegação, em Settings (Configurações), selecione SMS and voice (SMS e voz).
3. Na guia 10DLC campaigns (Campanhas 10DLC), escolha Register company (Inscrever empresa).

#### Note

A página Register your company (Inscrever empresa) exibe Resitration fee (Taxa de inscrição). Essa é uma taxa única associada ao registro da empresa. Esse custo é distinto de quaisquer outros custos ou taxas mensais. Ele é cobrado quando você inscreve sua empresa ou quando modifica os detalhes de um registro de empresa existente.

4. Na seção General info (Informações gerais), faça o seguinte:
  - Legal company name (Razão social da empresa): insira o nome sob o qual a empresa está registrada. O nome inserido deve corresponder exatamente ao nome da empresa associado ao ID fiscal fornecido.

 Important

Use a razão social exata de sua empresa. Depois de enviar, você não pode alterar essas informações. Informações incorretas ou incompletas podem resultar em atraso ou negação do registro.

- Em What type of legal form is this organization (Qual o tipo de forma jurídica dessa organização), escolha a opção que melhor descreva sua empresa.

 Note

As opções US government (Governo dos EUA) e Not-for-profit (Sem fins lucrativos) só podem ser usadas para inscrever organizações com sede nos Estados Unidos. Se sua organização estiver sediada em um país diferente dos EUA, você deve se inscrevê-la como Private for-profit (Privada com fins lucrativos), independentemente da forma jurídica real de sua organização.

- Se você escolheu Public for profit (Pública com fins lucrativos) na etapa anterior, insira o símbolo das ações da empresa e a bolsa de valores em que ela está listada.
- Escolha o país onde a empresa está registada em Country of registration (País de registro).
- Em Doing Business As (DBA) or brand name (Nome de fantasia ou nome da marca), insira outros nomes que sua empresa utiliza para fazer negócios.
- Em Tax ID (ID fiscal), insira o ID fiscal de sua empresa. O ID inserido depende do país em que sua empresa está registrada.
  - Se você estiver registrando uma entidade dos EUA ou de fora dos EUA que tenha um número de identificação do empregador (EIN) do IRS, insira seu EIN de nove dígitos. O nome jurídico da empresa, o EIN e o endereço físico inseridos devem corresponder às informações da empresa registradas no IRS.
  - Se você estiver registrando uma entidade canadense, insira seu número de corporação federal ou provincial. Não insira o número comercial (BN) fornecido pelo CRA. O nome jurídico da empresa, o número da corporação e o endereço físico inseridos devem corresponder às informações da empresa registradas na Corporations Canada.

- Se você estiver registrando uma entidade sediada em outro país, insira o ID fiscal principal de seu país. Em muitos países, essa é a parte numérica de seu número de ID de IVA.
  - Em Vertical, escolha a categoria que melhor descreva a empresa que você está registrando.
5. Na seção General info (Informações gerais), faça o seguinte:
- Em Address/Street (Endereço/Rua), insira o endereço físico associado à sua empresa.
  - Em City (Cidade), insira a cidade onde o endereço físico está localizado.
  - Em State or region (Estado ou região), insira o estado ou a região onde o endereço está localizado.
  - Em Zip Code/Postal Code (CEP/Código postal), insira o CEP ou código postal associado ao endereço.
  - Em Company website (Site da empresa), insira o URL completo do site de sua empresa. Inclua "http://" ou "https://" no início do endereço.
  - Em Support email (E-mail de suporte), insira seu endereço de e-mail.
  - Em Support phone number (Número de telefone de suporte), insira o código do país e o número do telefone.

 Note

O Campaign Registry requer um endereço de e-mail e número de telefone de contato caso eles precisem verificar as informações de registro junto a um representante de sua empresa.

6. Ao concluir, selecione Create (Criar). O registro de sua empresa é enviado ao The Campaign Registry. Na maioria dos casos, seu registro é aceito imediatamente e um status é fornecido.

Se o status do registro da sua empresa for Verified (Verificado), então você pode começar a criar campanhas 10DLC de baixo volume e uso misto. Você pode usar esse tipo de campanha para enviar até 75 mensagens por minuto para destinatários que usam a AT&T, e sua empresa registrada pode enviar 2 mil mensagens por dia para destinatários que usam a T-Mobile. Você também pode enviar mensagens para destinatários que usam outras operadoras dos EUA, como a Verizon e a US Cellular. Essas operadoras não aplicam estritamente os limites de throughput, mas monitoram intensamente as mensagens 10DLC em busca de sinais de spam e uso abusivo.

Se seu caso de uso exigir uma throughput que exceda esses valores, você poderá solicitar uma verificação adicional do registro de sua empresa. Para obter mais informações sobre como verificar seu registro de marca, consulte [Análise de seu registro 10DLC no Amazon SNS](#).

Se o status do registro de sua empresa for Unverified (Não verificado), houve problemas com as informações que você forneceu. Verifique as informações fornecidas e confirme se todos os campos contêm as informações corretas. Você pode fazer alterações em algumas partes do registro da empresa no console do Amazon Pinpoint. Para obter mais informações sobre como modificar detalhes de registro da empresa, consulte [Editar um registro de empresa 10DLC](#).

### Análise de seu registro 10DLC no Amazon SNS

Se o registro de sua empresa for bem-sucedido e você quiser inscrever uma campanha com recursos com throughput superior, será necessário analisar o registro da empresa.

Quando você analisa seu registro, uma organização terceirizada examina os detalhes da empresa que você forneceu e retorna uma pontuação de avaliação. Uma alta pontuação de avaliação pode resultar em throughput mais alto para sua empresa 10DLC e as campanhas associadas a ela. No entanto, a análise não garante o aumento do throughput.

As pontuações de avaliação não são aplicadas retroativamente. Em outras palavras, se você já criou uma campanha 10DLC e depois avaliou o registro de sua empresa, sua pontuação de avaliação não será aplicada automaticamente à uma campanha existente. Por esse motivo, você deve analisar sua empresa ou marca antes de criar qualquer campanha 10DLC.

#### Note

Há uma taxa não reembolsável de USD 40 para analisar sua empresa ou marca.

Para analisar o registro de sua empresa

1. Faça login no AWS Management Console e abra o console do Amazon Pinpoint em <https://console.aws.amazon.com/pinpoint/>.
2. No painel de navegação, em Settings (Configurações), selecione SMS and voice (SMS e voz).
3. Na guia 10DLC campaigns (Campanhas 10DLC), escolha a 10DLC company (Empresa 10DLC) que você deseja avaliar.
4. Na página de detalhes da empresa, na parte inferior, escolha Apply for vetting (Solicitar avaliação).

## 5. Na janela Apply for additional vetting (Solicitar análise adicional), escolha Submit (Enviar).

Para empresas sediadas nos EUA, o processo de análise normalmente leva cerca de um minuto para ser concluído. Para empresas não sediadas nos EUA, o processo de análise pode ser muito mais demorado, dependendo do tempo que leva para esses dados serem disponibilizado para o respectivo país.

Depois de enviar uma solicitação de avaliação, você retorna à página de detalhes da empresa. A seção Company vetting results (Resultados da avaliação da empresa) exibe o status e os resultados da solicitação de avaliação. Quando o processo de avaliação for concluído, esta tabela mostrará uma pontuação de avaliação na coluna Score (Pontuação). Sua pontuação de avaliação determina seus recursos de throughput de 10DLC. Seu throughput varia de acordo com o tipo de campanha criada por você. Se você criar campanhas 10DLC relacionadas a marketing ou de uso misto, precisará ter uma pontuação de avaliação maior do que seria necessário para outros tipos de campanha para alcançar throughput alto. Para obter mais informações sobre recursos dos números de telefone 10DLC, consulte [Recursos do 10DLC](#).

Se você alterar os detalhes do registro de sua empresa depois de concluir o processo de avaliação, poderá solicitar que seu registro seja examinado novamente. Se alterar apenas o Vertical para o registro de sua empresa, sua pontuação de avaliação não será alterada. Se alterar qualquer detalhe diferente de Vertical, o resultado da avaliação poderá mudar. Em ambos os casos, a taxa de avaliação única será cobrada novamente.

### Editar ou excluir uma empresa registrada

Você pode editar algumas das informações de registro 10DLC para sua empresa diretamente no console do Amazon Pinpoint. Você também pode excluir um registro de empresa 10DLC abrindo um caso no AWS Support Center.

### Editar um registro de empresa 10DLC

Depois de concluir o processo de registro 10DLC para uma empresa, você poderá editar os respectivos detalhes.

Se vir uma mensagem de erro depois de editar os detalhes de registro de sua empresa, deve haver outros problemas com seu registro. Você pode abrir um ticket com o AWS Support para solicitar mais informações.

## Como editar um registro de empresa

1. Abra o AWS SMS console em <https://console.aws.amazon.com/sms-voice/>.
2. Siga as instruções para [editar seu registro](#) no Guia do usuário do Amazon Pinpoint SMS.

## Excluir um registro de empresa 10DLC

### Para excluir o registro de uma empresa

1. Abra o AWS SMS console em <https://console.aws.amazon.com/sms-voice/>.
2. Siga as instruções para [excluir seu registro](#) no Guia do usuário do Amazon Pinpoint SMS.

## Registrar uma campanha 10DLC

Ao registrar uma campanha 10DLC, você fornece uma descrição de seu caso de uso, bem como os modelos de mensagens que pretende usar. Antes de criar e inscrever uma campanha 10DLC, você deve primeiro inscrever sua empresa. Para obter informações sobre o registro de uma empresa, consulte [Registrar uma empresa](#).

### Note

Depois de registrar sua empresa, o Amazon Pinpoint mostra um dos dois status para o registro: Verified (Verificado) ou Unverified (Não verificado). Você só pode concluir o processo de registro da campanha 10DLC se o status de sua empresa for Verified (Verificado). Você poderá criar campanhas de uso misto de baixo volume.

Se o status for Unverified (Não verificado), isso geralmente significa que alguns dos dados que você forneceu ao inscrever sua empresa estavam incorretos. Você não poderá criar nenhuma campanha 10DLC enquanto sua empresa tiver esse status. Você pode modificar o registro da empresa para tentar corrigir esses problemas. Para obter mais informações sobre como modificar os registros de empresa 10DLC, consulte [Editar ou excluir uma empresa registrada](#).

Nesta página, primeiro você fornece os detalhes da empresa para a qual está criando a campanha 10DLC e, em seguida, fornece os detalhes do caso de uso da própria campanha. As informações nesta página são fornecidas ao The Campaign Registry para aprovação.

Nesta seção, você escolherá a empresa para a qual está criando a campanha 10DLC e fornecerá detalhes adicionais.

Para inscrever uma campanha 10DLC

1. Faça login no AWS Management Console e abra o console do Amazon Pinpoint em <https://console.aws.amazon.com/pinpoint/>.
2. Em SMS and voice (SMS e voz), escolha Phone numbers (Números de telefone).
3. Na guia 10DLC campaigns (Campanhas 10DLC), escolha Create a 10DLC campaign (Criar uma campanha 10DLC).
4. Na página Create a 10DLC campaign (Criar uma campanha 10DLC), na seção Campaign info (Informações da campanha), faça o seguinte:
  - a. Em Company name (Nome da empresa), escolha a empresa para a qual você está criando a campanha. Se você ainda não tiver registrado a empresa, deve fazer isso primeiro. Para obter informações sobre o registro de uma empresa, consulte [Registrar uma empresa](#).
  - b. Em Campaign name (Nome da campanha), digite um nome para a campanha.
  - c. Em Vertical, escolha a opção que melhor represente sua empresa.
  - d. Em Help message (Mensagem de ajuda), insira a mensagem que seus clientes receberão se enviarem a palavra-chave “HELP” para seu número de telefone 10DLC.
  - e. Em Stop message (Mensagem de interrupção), insira a mensagem que seus clientes receberão se enviarem a palavra-chave “STOP” para seu número de telefone 10DLC.

 Tip

Seus clientes podem responder às suas mensagens com a palavra “HELP” para saber mais sobre as mensagens que estão recebendo de você. Eles também podem responder “STOP” para optar por não receber mais suas mensagens. As operadoras de telefonia móvel dos EUA exigem que você forneça respostas a ambas as palavras-chave.

Veja a seguir um exemplo de uma resposta HELP que atende aos requisitos das operadoras móveis dos EUA:

**ExampleCorp Account Alerts: For help call 1-888-555-0142 or go to example.com. Msg&data rates may apply. Text STOP to cancel.**

Veja a seguir um exemplo de resposta STOP compatível.

**You are unsubscribed from ExampleCorp Account Alerts. No more messages will be sent. Reply HELP for help or call 1-888-555-0142.**

Suas respostas a essas palavras-chave devem conter 160 caracteres ou menos.

5. Na seção Campaign use case (Caso de uso da campanha), faça o seguinte:
  - a. Em Use case type (Tipo de caso de uso), se você tiver um caso de uso relacionado a caridade, escolha Special (Especial). Do contrário, escolha Standard (Padrão).
  - b. Em Use case (Caso de uso), escolha um caso de uso que mais se assemelhe à sua campanha na lista predefinida de casos de uso. A taxa mensal de cada caso de uso aparece ao lado do nome do caso de uso.

 Note

A cobrança mensal para registrar a campanha 10DLC é mostrada ao lado de cada tipo de caso de uso. A maioria dos tipos de campanha 10DLC tem a mesma cobrança mensal. A cobrança pelo registro de casos de uso mistos de baixo volume é menor do que para outros tipos de casos de uso. No entanto, as campanhas mistas de baixo volume comportam taxas de transferência mais baixas do que outros tipos de campanha.

- c. Introduza pelo menos uma mensagem de SMS de exemplo em Sample SMS message (Exemplo de mensagem SMS). Essa é a mensagem de exemplo que você planeja enviar aos seus clientes. Se você pretende usar vários modelos de mensagens para essa campanha 10DLC, inclua-os também.

 Important

Não use texto de espaço reservado para suas mensagens de exemplo. As mensagens de exemplo fornecidas devem refletir, com a maior precisão possível, as mensagens reais que você pretende enviar.

6. A seção Campaign and content attributes (Atributos de campanha e conteúdo) contém uma série de perguntas com Yes (Sim) ou No (Não) relacionadas aos recursos específicos da campanha. Alguns atributos são obrigatórios, portanto, não é possível alterar o valor padrão.

Os atributos escolhidos devem ser aplicáveis à sua campanha.

Indique se cada uma das opções a seguir se aplica à campanha que você está inscrevendo:

- **Subscriber opt-in (Inclusão de assinantes):** os assinantes podem optar por receber mensagens sobre esta campanha.
- **Subscriber opt-out (Exclusão de assinantes):** os assinantes podem optar por deixar de receber mensagens sobre esta campanha.
- **Subscriber help (Ajuda do assinante):** os assinantes podem contatar o remetente da mensagem depois de enviar a palavra-chave HELP.
- **Number pooling (Agrupamento de números):** essa campanha 10DLC usa mais de 50 números de telefone.
- **Direct lending or loan arrangement (Contratação de empréstimo ou empréstimo direto):** a campanha inclui informações sobre empréstimos diretos ou outros acordos de empréstimo.
- **Embedded link (Link incorporado):** a campanha 10DLC inclui um link incorporado. Links de encurtadores de URL comuns, como TinyUrl ou Bitly, não são permitidos. No entanto, você pode usar encurtadores de URL que oferecem domínios personalizados.
- **Embedded phone number (Número de telefone incorporado):** a campanha inclui um número de telefone incorporado que não é um número de suporte ao cliente.
- **Affiliate marketing (Marketing de afiliados):** a campanha 10DLC inclui informações de marketing de afiliados.
- **Age-gated content (Conteúdo limitado por idade):** a campanha 10DLC inclui conteúdo com idade limitada, conforme definido pelas diretrizes da operadora e da Cellular Telecommunications and Internet Association (CTIA).

## 7. Escolha Create (Criar).

Depois de enviar os detalhes de registro de sua campanha, a página SMS and voice (SMS e voz) será exibida. Uma mensagem é exibida indicando que sua campanha foi enviada e está sendo revisada. É possível ver o status de sua solicitação na guia 10DLC campaigns (Campanhas 10DLC). É possível verificar o status de sua inscrição na guia 10DLC, que será uma das opções seguintes:

- **Active (Ativo):** sua campanha 10DLC foi aprovada. É possível solicitar um 10DLC e atribuir esse número à sua campanha. Para obter mais informações, consulte [Solicitar números 10DLC, números de chamada gratuita e códigos longos P2P para mensagens SMS com o Amazon SNS](#).

- **Pending (Pendente):** sua campanha 10DLC ainda não foi aprovada. Em alguns casos, a aprovação pode levar uma semana ou mais. Se o status sofrer alguma alteração, o console do Amazon Pinpoint refletirá essa condição. Não notificamos você sobre alterações de status.
  - **Rejected (Rejeitada):** sua campanha 10DLC foi rejeitada. Para obter mais informações, envie uma solicitação de suporte que inclua o ID da campanha rejeitada.
  - **Suspended (Suspenso):** uma ou mais operadoras suspenderam sua campanha 10DLC. Para obter mais informações, envie uma solicitação de suporte que inclua o ID da campanha suspensa. O Amazon Pinpoint não inclui motivos de suspensão no console e não notificamos você se sua campanha for suspensa.
8. Se seu 10DLC for aprovado, será possível solicitar um número 10DLC para se associar a essa campanha. Para obter informações sobre solicitar um número 10DLC, consulte [Solicitar números 10DLC, números de chamada gratuita e códigos longos P2P para mensagens SMS com o Amazon SNS](#).

## Utilização de campanhas 10DLC em várias regiões da AWS

Quando você inscreve uma empresa, ela fica disponível em sua conta da AWS em todas as regiões da AWS. No entanto, o mesmo não se aplica às campanhas 10DLC. Uma campanha 10DLC só pode ser usada na região da AWS em que foi inscrita.

Se você pretende usar 10DLC em mais de uma região da AWS, deverá registrar campanhas 10DLC separadas em cada uma dessas regiões. Essa etapa é necessária para cumprir os requisitos da operadora. Você é cobrado por campanha inscrita, mesmo que o caso de uso seja exatamente o mesmo.

A inscrição de várias campanhas oferece o benefício adicional de aumentar suas taxas de transferência para mensagens enviadas a destinatários que usam a AT&T como operadora de celular, porque a AT&T fornece taxas de transferência de 10DLC para cada campanha. Compare isso com a forma como a T-Mobile lida com a taxa de transferência de 10DLC, que se baseia em uma alocação diária de mensagens para cada empresa (independentemente do número de campanhas).

## Editar ou excluir uma campanha 10DLC

Você pode editar a resposta HELP, a resposta STOP e as mensagens de exemplo de uma campanha 10DLC usando o console do Amazon Pinpoint. Você também pode excluir campanhas 10DLC usando o console.

## Editar uma campanha 10DLC

Depois que sua campanha for aprovada, você poderá modificar as mensagens HELP, STOP e de exemplo. Você também pode adicionar outras mensagens de exemplo. As alterações nesses campos não exigem nova aprovação do Campaign Registry ou das operadoras. Você não pode modificar nenhum outro campo depois que a campanha 10DLC for aprovada.

Você pode ter no máximo cinco mensagens de exemplo. Não é possível reduzir o número de mensagens de exemplo que você registrou originalmente. Por exemplo, se você registrou sua campanha com três mensagens SMS de exemplo, não poderá reduzir o número de mensagens SMS de exemplo para menos de três.

### Note

Se quiser modificar quaisquer campos que não sejam as mensagens HELP, STOP e de exemplo, você deverá primeiro excluir a campanha 10DLC e recriá-la para incluir as informações atualizadas.

## Para editar uma campanha 10DLC

1. Faça login no AWS Management Console e abra o console do Amazon Pinpoint em <https://console.aws.amazon.com/pinpoint/>.
2. No painel de navegação, em Settings (Configurações), selecione SMS and voice (SMS e voz).
3. Na página 10DLC campaigns (Campanhas 10DLC), escolha a campanha que você deseja gerenciar.
4. Na seção Campaign messages (Mensagens de campanha) da página de detalhes da campanha, escolha Edit (Editar).
5. Atualize qualquer um dos seguintes campos:
  - Help message (Mensagem de ajuda)
  - Stop message (Mensagem de interrupção)
  - Sample SMS message (Exemplo de mensagem SMS)

Você não pode excluir uma mensagem de exemplo adicionada anteriormente nem o conteúdo de uma mensagem de exemplo para que o campo fique vazio. Se excluir o conteúdo de uma mensagem sem o substituir, a mensagem original será usada na atualização.

6. Escolha Update (Atualizar). Um banner de confirmação é exibido informando que as mensagens da campanha foram atualizadas.

## Excluir uma campanha 10DLC

Você pode excluir uma campanha 10DLC usando o console do Amazon Pinpoint. Antes de excluir uma campanha 10DLC, é necessário primeiro remover todos os números de telefone associados a essa campanha.

### Important

Se remover um número 10DLC de uma campanha, não terá mais acesso a ele. Além disso, as campanhas 10DLC excluídas não podem ser restauradas.

## Para editar ou excluir uma campanha 10DLC

1. Faça login no AWS Management Console e abra o console do Amazon Pinpoint em <https://console.aws.amazon.com/pinpoint/>.
2. No painel de navegação, em Settings (Configurações), selecione SMS and voice (SMS e voz).
3. Na página 10DLC campaigns (Campanhas 10DLC), escolha a campanha que você deseja gerenciar.
4. Na seção Phone numbers (Números de telefone), tome nota dos números de telefone associados à campanha.
5. Na guia Phone numbers (Números de telefone), escolha o número 10DLC que deseja remover e selecione Remove phone number (Remover número de telefone).

### Note

Essa etapa só é necessária se tiver vários números de telefone 10DLC associados à campanha. Se você tiver apenas um número de telefone associado à campanha 10DLC, ele aparecerá na guia 10DLC campaigns (Campanhas 10DLC). Tome nota do número exibido na guia.

6. Na caixa de confirmação, insira **delete** e escolha Confirm (Confirmar). Uma mensagem de êxito é exibida na parte superior da página SMS and voice (SMS e voz).
7. Repita as duas etapas anteriores para cada número 10DLC associado à campanha.

8. Depois de remover quaisquer números associados à campanha 10DLC, escolha a guia 10DLC campaigns (Campanhas 10DLC).
9. Escolha a campanha 10DLC que você quer modificar.
10. No canto superior direito da página 10DLC campaign details (Detalhes da campanha 10DLC), escolha Delete (Excluir).
11. Na caixa de confirmação, insira **delete** e escolha Confirm (Confirmar). Uma mensagem de êxito é exibida na parte superior da página SMS and voice (SMS e voz).

## Associar um código longo a uma campanha 10DLC

Se você já tem um código longo, pode associá-lo a uma de suas campanhas 10DLC atuais preenchendo uma solicitação de suporte. O código longo associado à campanha 10DLC pode ser usado somente com essa campanha e não pode ser usado para nenhuma outra campanha 10DLC. Você ainda poderá usar seu código longo enquanto ele estiver sendo migrado para o 10DLC. No entanto, até a aprovação, você não poderá usá-lo em nenhuma campanha de 10DLC.

Ao fazer a solicitação, você precisará:

- Dos códigos longos para associar a uma campanha 10DLC
- Do ID da campanha 10DLC para associar ao código longo

### Note

Essa campanha 10DLC precisa estar registrada antes de poder associar quaisquer códigos longos a ela. Se você ainda não criou e registrou uma campanha 10DLC, consulte [Registrar uma campanha 10DLC](#).

Para atribuir um código longo ao número 10DLC

1. Faça login no AWS Management Console e abra o console do Amazon Pinpoint em <https://console.aws.amazon.com/pinpoint/>.
2. Em Settings (Configurações) e, em seguida, em SMS and voice (SMS e voz), escolha a guia Phone numbers (Números de telefone).
3. Escolha o código longo que você deseja converter em 10DLC.

4. Para abrir a Central de Suporte, escolha Assign to 10DLC campaign (Atribuir à campanha 10DLC).
5. Para o tipo de caso, escolha Service limit increase (Aumento do limite de serviço).
6. Em Limit type (Tipo de limite), escolha Pinpoint (Identificar).
7. Na seção Requests (Solicitações), escolha a Region (Região) e, em seguida, em Limit (Limite), escolha 10 DLC – Associate existing US long code to 10DLC campaign (10 DLC – Associar código longo dos EUA existente à campanha 10DLC).
8. Em Case description (Descrição do caso), para Use case description (Descrição do caso de uso), certifique-se de incluir o ID da campanha 10DLC e os números de códigos longos que você deseja associar a campanha. É possível incluir vários códigos longos na solicitação, mas apenas um ID de campanha deve ser incluído.
9. Em Contact options (Opções de contato), em Preferred contact language (Idioma de contato preferido), escolha o idioma que você prefere usar para se comunicar com a equipe de suporte da AWS.
10. Em Contact method (Método de contato), escolha seu método preferido de comunicação com a equipe de suporte da AWS.
11. Escolha Submit (Enviar).

### Acesso entre contas do 10DLC

Cada número de telefone 10DLC está associado a uma única conta em uma única região da AWS. Se deseja usar o mesmo número de telefone 10DLC para enviar mensagens em mais de uma conta ou região, há duas opções:

1. Você pode registrar a mesma empresa e campanha em cada uma de suas contas da AWS. Esses registros são gerenciados e cobrados separadamente. Se você registrar a mesma empresa em várias contas da AWS, o número de mensagens que poderá enviar por dia aos clientes da T-Mobile será compartilhado entre cada uma dessas contas.
2. Você pode concluir o processo de registro 10DLC em uma conta da AWS e usar o AWS Identity and Access Management (IAM) para conceder permissão a outras contas para usar seu número 10DLC para envio.

#### Note

Essa opção realmente permite o acesso entre contas aos seus números de telefone 10DLC. No entanto, observe que as mensagens enviadas de suas contas secundárias

são tratadas como se tivessem sido enviadas de sua conta principal. As cotas e o faturamento são contabilizados nessa conta e não em contas secundárias.

## Configurar o acesso entre contas usando políticas do IAM

Você pode usar funções do IAM para associar outras contas à sua conta principal. Em seguida, você pode delegar permissões de acesso de sua conta principal para suas contas secundárias, concedendo-lhes acesso aos números 10DLC na conta principal.

### Para conceder acesso a um número 10DLC em sua conta principal

1. Se ainda não tiver feito isso, conclua o processo de registro 10DLC na conta principal. Esse processo envolve três etapas:
  - Registre sua empresa. Para obter mais informações, consulte [Inscrição de sua empresa ou marca](#) para uso com 10DLC.
  - Inscreva sua campanha 10DLC (caso de uso). Para obter mais informações, consulte [Registrar uma campanha 10DLC](#).
  - Associe um número de telefone à campanha 10DLC. Para obter mais informações, consulte [Associar um código longo a uma campanha 10DLC](#).
2. Crie uma função do IAM em sua conta principal que permita que outra conta chame a operação de API Publish para o número de telefone 10DLC. Para obter mais informações sobre como criar uma função do IAM, consulte [Criar funções do IAM](#) no Guia do usuário do IAM.
3. Delegue e teste a permissão de acesso de sua conta principal usando funções do IAM com qualquer uma de suas contas que precisem usar seus números 10DLC. Por exemplo, é possível delegar permissão de acesso de sua conta de produção para sua conta de implantação. Para obter mais informações sobre como delegar e testar permissões, consulte [Delegar acesso entre AWS contas usando funções do IAM](#) no Guia do usuário do IAM.
4. Usando a nova função, envie uma mensagem usando um número 10DLC da conta principal. Para obter mais informações sobre a utilização de uma função, consulte [Usar funções do IAM](#) no Guia do usuário do IAM.

## Obtenção de informações sobre problemas de registro de 10DLC

Em algumas situações, você pode receber uma mensagem de erro ao tentar inscrever sua empresa ou sua campanha 10DLC.

## Problemas de registro da empresa

Quando você inscreve sua empresa, são exibidos um dos dois status de registro: Verified (Verificado) ou Unverified (Não verificado). Se o status de registro da empresa for Verified (Verificado), então o registro foi bem-sucedido. Você pode começar a criar campanhas 10DLC.

Se o status do registro de sua empresa for Unverified (Não verificado), houve problemas com as informações que você forneceu. O console do Amazon Pinpoint fornece informações sobre os motivos pelos quais o registro da empresa recebeu esse status.

Para visualizar problemas de registro de sua empresa 10DLC

1. Faça login no AWS Management Console e abra o console do Amazon Pinpoint em <https://console.aws.amazon.com/pinpoint/>.
2. No painel de navegação, em SMS, selecione Números de telefone.
3. Na guia 10DLC campaigns (Campanhas 10DLC), na lista de campanhas, escolha o nome da empresa sobre a qual deseja encontrar mais informações.
4. A página de detalhes da empresa contém informações sobre os problemas identificados em seu registro. Se um campo na seção Company info (Informações da empresa) contiver um símbolo de aviso, o problema de registro está relacionado às informações nesse campo.

Verifique as informações fornecidas e confirme se todos os campos contêm as informações corretas. Você pode editar o registro da empresa no console do Amazon Pinpoint. Para obter mais informações sobre como modificar detalhes de registro da empresa, consulte [Editar ou excluir uma empresa registrada](#).

## Problemas de registro de campanha

Ao registrar sua campanha 10DLC, talvez você veja uma mensagem de erro em determinadas situações.

Se não conseguir identificar o problema referente a seu registro, você poderá abrir um caso junto ao [AWS Support Center](#) para solicitar outras informações. Use os procedimentos a seguir para criar um caso do AWS Support. A equipe do AWS Support fornecerá informações sobre os motivos pelos quais o registro da campanha 10DLC foi rejeitado.

Para enviar uma solicitação de informações sobre uma campanha 10DLC rejeitada

1. Faça login no AWS Management Console em <https://console.aws.amazon.com/>.

2. No menu Support (Suporte), escolha Support Center.
3. No painel Seus casos de suporte, escolha Criar caso.
4. Escolha o link Procurando aumentos no limite de serviço? e preencha o seguinte:
  - Em Tipo de limite, escolha Pinpoint SMS.
5. Em Requests (Solicitações), preencha as seguintes seções:
  - Em Região, escolha a Região da AWS na qual você tentou inscrever a campanha.

 Note

A região é obrigatória na seção Solicitações. Mesmo que você tenha fornecido essas informações na seção Detalhes do caso, também é necessário incluí-las aqui.

- Em Resource Type (Tipo de recurso), escolha 10DLC Registration (Registro 10DLC).
  - Em Limite, escolha Rejeição de registro de empresa ou campanha 10DLC.
6. Em Novo valor de limite, escolha o aumento do limite para o tipo de limite. Normalmente, esse valor é **1**.
  7. Em Descrição do caso, insira o ID de campanha 10DLC rejeitado.
  8. (Opcional) Se você quiser enviar outras solicitações, escolha Adicionar outra solicitação. Se você incluir várias solicitações, forneça as informações necessárias para cada uma delas. Para obter as informações necessárias, consulte as outras seções em [Solicitar suporte para sistema de mensagens SMS com o Amazon SNS](#).
  9. Em Opções de contato, para Idioma de contato preferencial, escolha o idioma no qual você deseja receber as comunicações sobre esse caso.
  10. Quando terminar, escolha Submit (Enviar).

## Números de chamada gratuita

Um número de chamadas gratuitas (TFN) é um número de 10 dígitos que começa com um dos seguintes códigos de área: 800, 888, 877, 866, 855, 844 ou 833. É possível usar TFNs para enviar apenas mensagens transacionais.

### Important

As operadoras de telefonia móvel dos EUA mudaram recentemente seus regulamentos e exigirão que todos os números de chamada gratuita (TFNs) conclua um processo de registro em um órgão regulador antes de 30 de setembro de 2022. Confira o status de seu TFN acessando [the section called “Status do registro de número gratuito”](#). Para obter informações sobre o registro de sua empresa, consulte [the section called “Registrar seu número de chamada gratuita”](#).

Após o envio, pode levar até 15 dias úteis para o processamento do registro.

Atualização em 3 de março de 2023: a partir de 1º de abril de 2023, as operadoras de celular aplicarão os seguintes limites em todo o setor para mensagens enviadas por qualquer número de chamada gratuita não registrado:

- Limite diário: 500 mensagens, reinicializado às 12:00 AM PT (16:00, hora de Brasília)
- Limite semanal: 1.000 mensagens, reinicializado no domingo às 12:00 AM PT (16:00, hora de Brasília)
- Limite mensal: 2.000 mensagens, reinicializando no final do mês civil às 12:00 AM PT (16:00, hora de Brasília)

Atualização em 19 de setembro de 2022: a partir de 1.º de outubro de 2022, as operadoras de celular passaram a aplicar os seguintes limites em todo o setor para mensagens enviadas por qualquer número de chamada gratuita não registrado:

- Limite diário: 2 mil mensagens
- Limite diário: 12 mil mensagens
- Limite diário: 25 mil mensagens

É altamente recomendável atualizar seu registro assim que possível. Mensagens enviadas por meio de TFNs não registrados terão como base o melhor esforço. As mensagens estarão sujeitas a uma maior filtragem e bloqueio ao longo do tempo, à medida que as operadoras continuarem restringindo o tráfego não registrado.

## Tópicos

- [Diretrizes para usar números de chamada gratuita](#)

- [Comprar um número de chamada gratuita](#)
- [Requisitos e processo de registro de número de chamada gratuita](#)
- [Status do registro de número gratuito](#)
- [Editando, descartando e excluindo seu registro](#)
- [Problemas de registro](#)
- [Perguntas frequentes sobre números de chamada gratuita](#)
- [Vantagens e desvantagens dos números de chamada gratuita](#)

## Diretrizes para usar números de chamada gratuita

Os TFNs normalmente são usados apenas nos EUA para mensagens transacionais, como confirmação de registro ou envio de senhas de uso único. Eles podem ser usados para mensagens de voz e SMS. O throughput médio é de três partes de mensagem por segundo (MPS). No entanto, esse throughput é afetado pela codificação de caracteres. Para obter mais informações sobre como a codificação de caracteres afeta partes da mensagem, consulte [Limites de caracteres de SMS no Amazon SNS](#). Para obter informações sobre o registro de um TFN, consulte [Requisitos e processo de registro de número de chamada gratuita](#).

Cada conta de cliente pode ter até cinco TFNs. Se você estiver enviando mais de 15 mensagens de texto por segundo, mas menos de 100, é recomendável registrar um ou mais [IDs de origem de 10DLC](#). Se seus casos de uso exigirem o envio de mais de 100 mensagens de texto por segundo, é recomendável comprar e registrar um ou mais [códigos curtos](#).

Ao usar um TFN como um número de origem, siga estas instruções:

- Não use URLs abreviados criados a partir de encurtadores de URL de terceiros, pois essas mensagens são mais propensas a serem filtradas como spam.

Se for necessário usar um URL abreviado, avalie a possibilidade de usar um [número 10DLC](#) ou [código curto](#). O uso de códigos curtos e 10DLCs exige que você registre seu modelo de mensagem, onde é possível especificar um URL abreviado.

- Lembre-se de que as respostas às palavras-chave de inclusão (STOP) e exclusão (UNSTOP) são definidas pela operadora. Não é possível modificar essas nem outras palavras-chave. Também não é possível modificar as mensagens enviadas quando os usuários respondem com STOP e UNSTOP.

- Não envie conteúdo de mensagem igual ou semelhante usando vários TFNs. As operadoras chamam essa prática de snowshoeing ou agrupamento de números e direciona essas mensagens para filtragem.
- As mensagens relacionadas aos seguintes setores podem ser consideradas restritas e estão sujeitas a filtragem pesada ou bloqueios definitivos. Isso pode incluir senhas de uso único (OTP) e autenticação multifator (MFA) para serviços relacionados a categorias restritas.

Se você teve um registro negado por ser um caso de uso não compatível e acha que essa designação está incorreta, pode enviar uma solicitação por meio do suporte. Para obter detalhes sobre como fazer isso, consulte [Problemas de registro](#).

A tabela a seguir descreve os tipos de conteúdo restrito:

Categoria	Exemplos
Jogos de aposta	<ul style="list-style-type: none"> <li>• Aplicações/sites</li> <li>• Cassinos</li> <li>• Sorteios</li> </ul>
Serviços financeiros de alto risco	<ul style="list-style-type: none"> <li>• Autoempréstimos</li> <li>• Criptomoedas</li> <li>• Cobrança de dívidas</li> <li>• Empréstimos consignados</li> <li>• Empréstimos de curto prazo com juros elevados</li> <li>• Empréstimos hipotecários</li> <li>• Empréstimos estudantis</li> <li>• Alertas de ações</li> </ul>
Perdão de dívidas	<ul style="list-style-type: none"> <li>• Consolidação de dívidas</li> <li>• Redução des dívida</li> <li>• Programas de restauração de crédito</li> </ul>
et-rich-quick Esquemas G	<ul style="list-style-type: none"> <li>• ork-from-home Programas W</li> <li>• Oportunidades de investimento de risco</li> </ul>

Categoria	Exemplos
	<ul style="list-style-type: none"><li>• Esquemas de pirâmide ou de marketing multinível</li></ul>
Substâncias proibidas/controladas	<ul style="list-style-type: none"><li>• Cannabis/CBD</li></ul>
Phishing	<ul style="list-style-type: none"><li>• Tentativas de fazer com que os usuários revelem informações pessoais ou informações de login de sites</li></ul>
S.H.A.F.T.	<ul style="list-style-type: none"><li>• Sexo</li><li>• Ódio</li><li>• Álcool</li><li>• Armas de fogo</li><li>• Tabaco/Vape</li></ul>

## Comprar um número de chamada gratuita

Para comprar TFNs, use o console do Amazon Pinpoint em <https://console.aws.amazon.com/sms-voice/>. Para ter mais informações, consulte [Requisitos e processo de registro de número de chamada gratuita](#).

Atualmente, o Amazon Pinpoint SMS oferece suporte a números gratuitos para mensagens de voz e SMS. O Amazon SNS aceita somente mensagens SMS.

## Requisitos e processo de registro de número de chamada gratuita

### Important

É possível revogar um TFN se ele for usado para qualquer outra finalidade que não seja o caso de uso especificado.

## Casos de uso proibidos com números de chamada gratuita

O Amazon SNS tem capacidade limitada de envio de mensagens nos casos em que as mensagens estão bloqueadas (por exemplo, casos de uso relacionados a substâncias controladas ou phishing)

ou quando altos níveis de filtragem são esperados (por exemplo, mensagens financeiras de alto risco). Talvez você não consiga registrar TFNs associados a casos de uso de conteúdo restrito definidos em [Diretrizes para usar números de chamada gratuita](#).

## Registrar seu número de chamada gratuita

Depois de comprar um TFN, você deve registrar o número. Para obter instruções sobre como fazer isso, consulte o [processo de registro de números gratuitos](#) no Guia do usuário de SMS do Amazon Pinpoint.

## Registro de autoatendimento para números gratuitos nas regiões de SMS do Amazon Pinpoint

Se você solicitou o TFN nas regiões de [SMS do Amazon Pinpoint](#), conclua o processo de registro da empresa diretamente no console de [SMS do Amazon Pinpoint usando as instruções encontradas no formulário de registro de números gratuitos dos EUA no Guia do usuário do Amazon Pinpoint](#) SMS.

Ao registrar o TFN, verifique se as informações estão completas e corretas ou seu registro poderá ser rejeitado. As informações inseridas devem corresponder exatamente à sede corporativa da empresa.

## Processo de registro manual baseado em formulário para números gratuitos em regiões que não sejam regiões de SMS do Amazon Pinpoint

1. Faça o download deste [US\\_TFN\\_Registration.zip](#) e use o exemplo de formulário de registro (AWS US Toll-Free Registration Form-Business - Final.docx) para preencher as informações necessárias no arquivo CSV de registro do TFN (BulkUSStFN - Final.csv).

Cada solicitação de registro ou caso de uso só pode ter até cinco TFNs. Se você acredita que se qualifica para uma isenção dessa regra, forneça uma explicação detalhada para ser avaliada. Liste todos os números de telefone associados ao registro ou ao caso de uso.

2. Crie um caso em [AWS Support](#). Anexe seu arquivo CSV preenchido ao caso e envie a solicitação de registro do TFN.
3. Escolha Criar caso e Procurando aumentos no limite de serviço?
4. No tipo de limite, escolha Mensagens de texto do SNS.
5. Em Resource Type (Tipo de recurso), escolha 10DLC or Toll-free number registration (Registro de 10DLC ou número de ligação gratuita).
6. Anexe o documento US\_TFN\_registration e envie a solicitação.

## Principal ponto a observar

1. Após o envio de todas as informações necessárias, os registros podem levar até duas semanas para serem processados. Se faltarem informações ou elas estiverem incompletas, o processo de registro será adiado. Se seu registro for rejeitado, ajudaremos você a descobrir o motivo pelo qual ele foi negado e sugeriremos métodos para melhorar sua campanha para que ela possa ser registrada.
2. Os TFNs funcionam bem para casos de uso transacionais, como autenticação multifator (MFA), em que um throughput limitado é necessário. Cada TFN pode enviar até três partes de mensagens de texto por segundo, e cada conta de cliente pode ter até cinco TFNs. Se você estiver enviando mais de 15 mensagens de texto por segundo, mas menos de 100, é recomendável registrar um ou mais IDs de origem de [10DLC](#). Se seus casos de uso exigirem o envio de mais de 100 mensagens de texto por segundo, é recomendável comprar e registrar um ou mais [códigos curtos](#). Para obter mais detalhes, consulte [Diretrizes para usar números de chamada gratuita](#).

## Status do registro de número gratuito

Para verificar o status do seu registro, consulte [Verificar o status do seu registro](#) no Guia do usuário do Amazon Pinpoint SMS.

## Editando, descartando e excluindo seu registro

Use o Guia do usuário do Amazon Pinpoint SMS para realizar as seguintes tarefas:

- [Edite seu registro](#)
- [Descarte seu registro](#)
- [Exclua seu registro](#)
- [Veja seus recursos de registro](#)

## Problemas de registro

Se o registro do seu número gratuito não for aceito, você receberá uma mensagem explicando o motivo da rejeição.

Como enviar uma solicitação de informações sobre um número gratuito rejeitado

1. Faça login AWS Management Console no <https://console.aws.amazon.com/>.

2. No menu Suporte, escolha Support Center.
3. No painel Seus casos de suporte, escolha Criar caso.
4. Escolha o link Procurando aumentos no limite de serviço? e preencha o seguinte:
  - Em Tipo de limite, selecione Pinpoint SMS.
5. Em Solicitações, preencha as seguintes seções:
  - Em Região, indique onde você tentou registrar a campanha.

 Note

A região é obrigatória na seção Solicitações. Mesmo que você tenha fornecido essas informações na seção Detalhes do caso, também é necessário incluí-las aqui.

- Em Tipo de recurso, escolha Registro de 10DLC ou TFN.
  - Em Limite, escolha Rejeição de registro de empresa ou campanha.
6. Em Novo valor de limite, escolha o aumento do limite para o tipo de limite. Normalmente, esse valor é **1**.
  7. (Opcional) Se você quiser enviar outras solicitações, escolha Adicionar outra solicitação. Para obter as informações necessárias, consulte as outras seções em [Solicitar suporte para sistema de mensagens SMS com o Amazon SNS](#).
  8. Em Descrição do caso de uso, insira o número gratuito rejeitado.
  9. Em Opções de contato, para Idioma de contato preferencial, escolha o idioma no qual você deseja receber as comunicações sobre esse caso.
  10. Quando terminar, escolha Enviar.

Perguntas frequentes sobre números de chamada gratuita

Perguntas frequentes sobre o processo de registro de números de TFN.

Tenho um número de chamada gratuita atualmente?

Como conferir se você tem um número de chamada gratuita

- [Abra o console Amazon Pinpoint SMS em https://console.aws.amazon.com/sms-voice/](https://console.aws.amazon.com/sms-voice/).
- No painel de navegação, em SMS, selecione Números de telefone.

- O type (tipo) do TFN está listado como toll-free (chamada gratuita).

Preciso registrar meu número de chamada gratuita?

Sim. Para continuar usando um TFN que você possui atualmente, é necessário registrá-lo antes de 30 de setembro de 2022. Se você comprar um novo TFN após 30 de setembro de 2022, deverá registrá-lo primeiro para que possa enviar mensagens.

Como faço para comprar um número de chamada gratuita?

Siga as instruções em [Solicitar um número de telefone usando o console Amazon Pinpoint SMS](#) para comprar um TFN.

Como faço para registrar meu número de chamada gratuita?

Siga as instruções em [the section called “Registrar seu número de chamada gratuita”](#) para registrar um TFN.

Qual é o status de registro do meu número de chamada gratuita e o que ele significa?

Siga as instruções em [the section called “Status do registro de número gratuito”](#) para conferir seu registro e status.

Quais informações preciso fornecer?

Você precisará fornecer o endereço de sua empresa, um contato comercial e um caso de uso para o TFN. Você pode encontrar as informações necessárias em [the section called “Registrar seu número de chamada gratuita”](#).

E se meu registro for rejeitado?

Se seu registro for rejeitado, o status será alterado para Requires Updates (Requer atualizações). Para fazer atualizações, consulte [the section called “Editando, descartando e excluindo seu registro”](#).

De quais permissões eu preciso?

O usuário/função do IAM que você usa para visitar o console de SMS do Amazon Pinpoint deve ter a **permissão “*sms-voice: \**”**, caso contrário, você receberá um erro de acesso negado.

Vantagens e desvantagens dos números de chamada gratuita

Vantagens

Os originadores gratuitos têm MPS mais alto em relação aos códigos longos, além de boa capacidade de entrega.

## Desvantagens

Não há controle sobre opções de cancelamento e consentimento, pois elas são gerenciadas pela operadora.

Não inclua URLs curtos em sua mensagem nem use o número para enviar mensagens promocionais. Em vez disso, use um número 10DLC ou um código curto. Ao usar um código curto ou um número 10DLC, você precisa registrar seus modelos de mensagem, que podem conter URLs curtos e podem ser mensagens promocionais. Para obter mais informações sobre códigos curtos, consulte [Códigos curtos](#). Para obter mais informações sobre 10DLC, consulte [10DLC](#).

## Códigos curtos

Os códigos curtos são sequências numéricas menores que um número de telefone comum. Por exemplo, nos Estados Unidos e no Canadá, os números de telefone padrão (códigos longos) contêm 11 dígitos; e os curtos, cinco ou seis dígitos. O Amazon SNS oferece suporte a códigos curtos dedicados.

## Códigos curtos dedicados

Se você envia um grande volume de mensagens SMS para destinatários nos Estados Unidos ou Canadá, pode comprar um código curto dedicado. Ao contrário dos códigos curtos no grupo compartilhado, os códigos curtos dedicados são reservados para seu uso exclusivo.

## Vantagens

O uso de um código curto memorável pode ajudar a criar confiança. Se você precisa enviar informações confidenciais, como senhas, é uma boa ideia enviá-las usando um código curto para que seu cliente possa rapidamente determinar se a mensagem realmente foi enviada de sua parte.

Se você estiver lançando uma nova campanha de aquisição de clientes, poderá convidar clientes em potencial a enviar uma palavra-chave ao seu código curto (por exemplo, “Envie ‘FUTEBOL’ para 10987 para receber notícias e informações sobre futebol”). Os códigos curtos são mais fáceis de lembrar que os códigos longos, e é mais fácil para os clientes inserir códigos curtos em seus dispositivos. Ao reduzir o número de dificuldades que os clientes encontram ao se cadastrarem em seus programas de marketing, você pode aumentar a eficácia de suas campanhas.

Como as operadoras de celular precisam aprovar os códigos curtos novos antes de ativá-los, é menos provável que elas marquem as mensagens enviadas de códigos curtos como não solicitadas.

Ao usar códigos curtos para enviar mensagens SMS, você pode enviar um maior volume de mensagens por período de 24 horas do que é possível com outros tipos de identidade de origem. Em outras palavras, você tem uma cota de envio muito maior. Você também pode enviar um volume muito maior de mensagens por segundo. Ou seja, você tem uma taxa de envio muito maior.

## Desvantagens

Não há custos adicionais para adquirir códigos curtos, e eles podem levar muito tempo para implementar. Por exemplo, nos Estados Unidos, há uma taxa única de configuração de \$650.00 (USD) para cada código curto, além de uma cobrança recorrente adicional de \$995.00 por mês para cada código curto. A ativação de um código curto pode levar entre 8 e 12 semanas em todas as redes de operadoras. Para descobrir o preço e o tempo de provisionamento para um país ou região diferente, conclua o procedimento descrito em [Solicitar códigos curtos dedicados para o sistema de mensagens por SMS com o Amazon SNS](#).

## Códigos longos de pessoa a pessoa (P2P)

### Important

Em 31 de agosto de 2023, um número dedicado, como um número [10DLC](#), ou um [número gratuito](#) passou a ser necessário para enviar mensagens de texto SMS para os Estados Unidos e os respectivos territórios (Porto Rico, Guam, Ilhas Samoa Americana e Ilhas Virgens dos US). Sua solicitação de código longo será rejeitada se você usar os Estados Unidos como local para essas regiões.

### Important

Desde 1º de junho de 2021, os provedores de telecomunicações dos EUA não oferecem mais suporte ao uso de códigos longos de pessoa a pessoa (P2P) para comunicações de aplicação a pessoa (A2P) para destinos dos EUA. Em vez disso, você precisa usar outro tipo de ID de origem para essas mensagens. Para obter mais informações, consulte [10DLC](#).

Os códigos longos P2P são números de telefone que usam o formato numérico do país ou região em que os destinatários estão localizados. Os códigos longos P2P também são conhecidos como

números longos ou números móveis virtuais. Por exemplo, nos Estados Unidos e Canadá, os códigos longos P2P contêm 11 dígitos: o número 1 (o código do país), um código de área de três dígitos e um número de telefone de sete dígitos.

Para obter mais informações sobre como solicitar códigos longos P2P, consulte [Solicitar números 10DLC, números de chamada gratuita e códigos longos P2P para mensagens SMS com o Amazon SNS](#).

## Vantagens

Os códigos longos P2P dedicados são reservados para serem usados apenas por sua conta do Amazon SNS, pois eles não são compartilhados com outros usuários. Quando você usa códigos longos P2P dedicados, pode especificar qual código longo P2P deseja usar ao enviar cada mensagem. Ao enviar várias mensagens para o mesmo cliente, você pode garantir que cada mensagem pareça estar sendo enviada do mesmo número de telefone. Por esse motivo, os códigos longos P2P dedicados podem ser úteis na criação de sua marca ou identidade.

## Desvantagens

Os códigos longos P2P não são compatíveis com comunicações A2P para destinos dos US.

Se você enviar várias centenas de mensagens por dia do mesmo código longo P2P dedicado, as operadoras de celular poderão identificá-lo como um número que envia mensagens não solicitadas. Se o seu código longo P2P for marcado, suas mensagens podem não ser entregues aos destinatários.

Os códigos longos P2P também têm taxas de transferência limitadas. As tarifas máximas de envio variam de acordo com o país. Entre em contato com o AWS Support para obter mais informações. Se você planeja enviar grandes volumes de mensagens SMS ou pretende enviar em uma taxa maior do que uma mensagem por segundo, deve comprar um código curto dedicado.

Algumas operadoras não permitem que você use códigos longos P2P para enviar mensagens SMS A2P, inclusive nos EUA. Um SMS A2P é uma mensagem enviada para um dispositivo móvel do cliente quando esse cliente envia seu número de telefone celular para um aplicativo. As mensagens A2P são conversas unidirecionais, como mensagens de marketing, senhas únicas e lembretes de compromisso. Se você planeja enviar mensagens A2P, deve comprar um código curto dedicado (se seus clientes estão nos Estados Unidos ou Canadá) ou usar um ID do remetente (se seus destinatários estão em um país ou região onde os IDs de remetente são compatíveis).

Um número 10DLC é usado apenas para enviar mensagens nos EUA. O uso de um número 10DLC exige que você registre a marca de sua empresa e a campanha à qual deseja associar o número.

Depois de aprovado, é possível solicitar um 10DLC na página SMS and voice (SMS e voz) no console do Amazon Pinpoint em <https://console.aws.amazon.com/pinpoint/>. Depois de solicitado, o tempo para receber a aprovação é de sete a dez dias. O número não pode ser usado com nenhuma outra campanha.

### Comparação entre números de produtos dos EUA

Esta tabela mostra a comparação de suporte para os tipos de número de telefone dos EUA.

Recurso do produto	Código curto	Número de chamada gratuita	10DLC
Formato do número	5 a 6 dígitos	Número de 10 dígitos	Número de 10 dígitos
Canal de suporte	SMS	SMS	SMS
Tipo de tráfego de SMS	Promocional e transacional	Transacional	Promocional e transacional
Requer aprovação	Sim	Não	Sim
Tempo estimado de provisionamento	12 semanas <sup>1</sup>	15 dias úteis	1 semana
Taxa de transferência de SMS (número de mensagens SMS por segundo) <sup>2</sup>	100 partes de mensagem por segundo, maior taxa de transferência disponível por uma taxa adicional.	3 partes de mensagem por segundo	Varia com base em seu registro 10DLC. Oferece suporte para até 100 partes de mensagem por segundo.
Palavras-chave obrigatórias	Inclusão, exclusão e HELP	STOP, UNSTOP. Elas são gerenciadas	Inclusão, exclusão e HELP

Recurso do produto	Código curto	Número de chamada gratuita	10DLC	
		as por rede. Não é possível alterar a opção de não receber mensagens e optar novamente		

<sup>1</sup> A estimativa de provisionamento não inclui o tempo de aprovação.

<sup>2</sup> Para obter mais informações sobre o tamanho máximo das mensagens SMS, consulte [Publicar em um telefone celular](#).

## Solicitar suporte para sistema de mensagens SMS com o Amazon SNS

Algumas opções de SMS com o Amazon SNS não estarão disponíveis em sua conta da AWS até entrar em contato com o AWS Support. Crie um caso na [AWS Support Center](#) para solicitar qualquer um dos itens a seguir:

- Um aumento de seu limite mensal de gastos de SMS

Por padrão, o limite de gastos mensal é 1,00 USD. O limite de gastos determina o volume de mensagens que é possível enviar com o Amazon SNS. Você pode solicitar um limite de gastos que atenda ao volume mensal de mensagens esperado para seu caso de uso de SMS.

- Uma migração da [sandbox de SMS](#) para que você possa enviar mensagens SMS sem restrições. Para obter mais informações, consulte [Sair da sandbox de SMS](#).
- Um [número de origem](#) dedicado
- Um ID do remetente dedicado

Um ID do remetente é um ID personalizado mostrado como remetente no dispositivo do destinatário. Por exemplo, é possível usar a marca de sua empresa para tornar a origem da mensagem mais fácil de reconhecer. O suporte a IDs de remetentes varia de acordo com o país ou a região. Para obter mais informações, consulte [Países e regiões compatíveis](#).

Ao criar seu caso na AWS Support Center, certifique-se de incluir todas as informações necessárias referentes ao tipo de solicitação que você está enviando. Caso contrário, o AWS Support deve entrar em contato com você para obter essas informações antes de prosseguir. Ao enviar um caso detalhado, você ajuda a garantir que seu caso seja realizado sem atrasos. Para ver os detalhes necessários para tipos específicos de solicitações de SMS, consulte os tópicos a seguir.

## Tópicos

- [Solicitar códigos curtos dedicados para o sistema de mensagens por SMS com o Amazon SNS](#)
- [Solicitar números 10DLC, números de chamada gratuita e códigos longos P2P para mensagens SMS com o Amazon SNS](#)
- [Solicitar IDs do remetente para sistema de mensagens por SMS com o Amazon SNS](#)
- [Solicitar aumentos de sua cota de gastos mensais de SMS para o Amazon SNS](#)

## Solicitar códigos curtos dedicados para o sistema de mensagens por SMS com o Amazon SNS

Um código simplificado é um número que você pode usar para enviar mensagens SMS de alto volume. Geralmente, os códigos simplificados são usados para mensagens de aplicativo para pessoa (A2P), autenticação de dois fatores (2FA) e marketing. Um código simplificado, em geral, contém entre três e sete dígitos, dependendo do país ou da região em que ele está baseado.

Você só pode usar códigos simplificados para enviar mensagens para os destinatários no mesmo país em que o código simplificado está baseado. Se seu caso de uso exigir que você use códigos simplificados em mais de um país, solicite um código curto separado para cada país onde seus destinatários estão localizados.

Para obter informações sobre a definição de preço dos códigos curtos, consulte [Amazon SNS pricing](#) (“Preço do Amazon SNS”).

### Important

Se você for novo em relação ao sistema de mensagens por SMS com o Amazon SNS, solicite um limite mensal de gastos de SMS que atenda às demandas de SMS esperadas de seu caso de uso. Por padrão, o limite de gastos mensal é 1,00 USD. Você pode solicitar um aumento do limite de gastos no mesmo caso de suporte que inclui sua solicitação de um código simplificado. Você também pode usar um caso separado. Para obter mais

informações, consulte [Solicitar aumentos de sua cota de gastos mensais de SMS para o Amazon SNS](#).

Além disso, se você estiver solicitando um código curto dedicado para enviar mensagens que contenham ou possam conter informações de saúde protegidas (PHI – Protected Health Information), é necessário identificar essa finalidade na Case description (Descrição de caso) quando você abrir um caso de suporte, conforme detalhado abaixo.

## Abrir um caso de suporte de código curto do Amazon SNS

Abra um caso com o AWS Support realizando as etapas a seguir.

### Note

Alguns campos no formulário de solicitação são marcados como "opcionais". No entanto, o AWS Support exige todas as informações mencionadas nas etapas a seguir para processar sua solicitação. Se você não fornecer todas as informações necessárias, poderá haver atrasos no processamento de sua solicitação.

Para solicitar um código simplificado dedicado

1. Visite a [AWS Support Center](#).
2. Faça login no AWS Management Console em <https://console.aws.amazon.com/>.
3. No menu Support (Suporte), escolha Support Center.
4. No painel Seus casos de suporte, escolha Criar caso.
5. Escolha o link Procurando aumentos no limite de serviço? e preencha o seguinte:
  - Em Tipo de limite, escolha Mensagens de texto do SNS e preencha o seguinte:
  - (Opcional) Em Fornecer um link ao site ou à aplicação que enviará mensagens SMS, forneça um link para o site ou o nome da aplicação em que os membros do público optam por receber suas mensagens SMS.
  - (Opcional) Em Qual tipo de mensagem você pretende enviar, escolha o tipo de mensagem que pretende enviar utilizando o código longo:
    - One Time Passwords (Senhas de uso único): mensagens que fornecem senhas que seus clientes usam para se autenticarem em seu site ou aplicação.

- Promotional (Promocional): mensagens não críticas que promovem o seu negócio ou serviço, como ofertas especiais ou anúncios.
  - Transactional (Transacional): mensagens informativas importantes que oferecem suporte para transações do cliente, como confirmações de pedidos ou alertas de contas. As mensagens transacionais não devem conter conteúdo promocional nem de marketing.
  - (Opcional) Em De qual região da AWS você enviará mensagens, escolha a região de onde você enviará mensagens.
  - (Opcional) Em Para quais países você pretende enviar mensagens, insira os países ou as regiões para os quais você pretende enviar mensagens SMS.
  - (Opcional) Em Como seus clientes optam por receber suas mensagens, forneça uma descrição de como seus clientes optam por receber suas mensagens.
  - (Opcional) No campo Forneça o modelo de mensagem que você planeja usar para enviar mensagens aos seus clientes, inclua o modelo que será usado.
6. Em Requests (Solicitações), preencha as seguintes seções:
- Em Região, escolha a Região da AWS para sua solicitação de código simplificado.

 Note

A região é obrigatória na seção Solicitações. Mesmo que você tenha fornecido essas informações na seção Detalhes do caso, também é necessário incluí-las aqui.

- Em Tipo de recurso, selecione Códigos SMS simplificados dedicados.
  - Em Limit (Limite), selecione a opção mais adequada ao seu caso de uso.
7. Em Novo valor de limite, selecione o número de IDs de remetente que você está solicitando. Normalmente, esse valor é **1**.
8. (Opcional) Se você quiser enviar outras solicitações, escolha Adicionar outra solicitação. Para obter as informações necessárias, consulte as outras seções em [Solicitar suporte para sistema de mensagens SMS com o Amazon SNS](#).
9. Em Descrição do caso, resuma o caso de uso e inclua como os destinatários realizarão o cadastro para as mensagens enviadas com o código simplificado, e forneça as seguintes informações:
- Informações da empresa:

- Company name (Nome da empresa)
- Endereço para correspondência da empresa
- Nome e número de telefone do contato primário para sua solicitação
- Endereço de e-mail e número gratuito para oferecer suporte à sua empresa
- Número de identificação fiscal da empresa
- Nome do seu produto ou serviço
  
- Processo de cadastro do usuário:
  - Site da empresa ou o site no qual seus clientes poderão se cadastrar para receber mensagens do seu código simplificado.
  - Como os usuários se cadastrarão para receber mensagens do seu código simplificado. Especifique uma ou mais das opções a seguir:
    - **Text messages**
    - **Website**
    - **Mobile app**
    - **Other** Se houver outras, explique.
  - O texto para a opção para se cadastrar em mensagens no seu site, aplicativo ou em outro lugar.
  - A sequência de mensagens que você planeja usar para inclusão dupla. Forneça todas as informações a seguir:
    1. A mensagem SMS que você planeja enviar quando um usuário se cadastrar. Essa mensagem solicita o consentimento do usuário para mensagens recorrentes. Por exemplo: ExampleCorp: Responda SIM para receber alertas de transações de conta. Podem se aplicar taxas de mensagens e dados.
    2. A resposta para inclusão que você espera do usuário. Geralmente é uma palavra-chave, como SIM.
    3. A mensagem de confirmação que você deseja enviar quando os clientes enviam essa palavra-chave para seu código simplificado. Por exemplo: Agora você está registrado para receber alertas de conta da ExampleCorp. Podem se aplicar taxas de mensagens e dados. Envie PARAR para cancelar ou AJUDA para receber informações.
  
- A finalidade de suas mensagens:

- A finalidade das mensagens que você planeja enviar com seu código simplificado. Especifique uma das seguintes opções:
  - **Promotions and marketing**
  - **Location-based services**
  - **Notifications**
  - **Information on demand**
  - **Group chat**
  - **Two-factor authentication (2FA)**
  - **Polling and surveys**
  - **Sweepstakes or contests**
  - **Other** Se houver outras, explique.
- Se você planeja usar seu código simplificado para enviar mensagens promocionais ou de marketing para uma empresa que não seja a sua.
- Se você planeja usar o código curto para enviar mensagens que contenham ou possam conter informações de saúde protegidas (PHI), conforme definido pela lei de portabilidade e responsabilidade de seguro saúde (HIPAA) e legislação e regulamentos associados.
- Conteúdo da mensagem:
  - A mensagem que você planeja enviar quando os clientes decidirem receber suas mensagens ao enviar uma palavra-chave específica. Tenha cuidado ao especificar essa palavra-chave e mensagem. Pode demorar várias semanas para alterar a mensagem. Quando criamos o seu código simplificado, registramos a palavra-chave e a mensagem com as operadoras de celular no país em que você usa o código simplificado. Sua mensagem pode se parecer com o seguinte exemplo: Bem-vindo aos alertas de *ProductName*! Taxas de mensagens e dados são aplicáveis. 2mensagens por mês. Responda AJUDA para obter ajuda; PARAR para cancelar.
  - A resposta que você deseja enviar quando os clientes responderem às suas mensagens com a palavra-chave HELP (Ajuda). Essa mensagem precisa incluir informações de contato de suporte ao cliente. Por exemplo: Alertas do *ProductName*: ajuda em *example.com/help* ou *(800) 555-0199*. Taxas de mensagens e dados são aplicáveis. 2mensagens por mês. Responda PARAR para cancelar.
  - A resposta que você deseja enviar quando os clientes responderem às suas mensagens com a palavra-chave STOP (Ajuda). Essa mensagem precisa confirmar que o usuário não

receberá mais mensagens de você. Por exemplo: Você cancelou a assinatura dos alertas do *ProductName*. Mensagens não serão mais enviadas. Responda AJUDA para obter ajuda ou (800) 555-0199.

- O texto que você planeja enviar como um lembrete periódico de que o usuário está inscrito para receber suas mensagens. Por exemplo: Lembrete: você está inscrito para receber alertas de conta da ExampleCorp. Podem se aplicar taxas de mensagens e dados. Envie PARAR para cancelar ou AJUDA para receber informações.
- Um exemplo de cada tipo de mensagem que você planeja enviar usando seu código simplificado. Forneça pelo menos três exemplos. Se você planeja enviar mais de três tipos de mensagens, forneça exemplos para todos eles.

 Important

Operadoras de celular exigem que forneçamos todas as informações listadas acima para provisionar códigos simplificados. Não será possível processar sua solicitação até que você forneça todas essas informações.

10. (Opcional) Se você quiser enviar outras solicitações, escolha Adicionar outra solicitação. Se você incluir várias solicitações, forneça as informações necessárias para cada uma delas. Para obter as informações necessárias, consulte as outras seções em [Solicitar suporte para sistema de mensagens SMS com o Amazon SNS](#).
11. Em Opções de contato, para Idioma de contato preferencial, escolha o idioma no qual você deseja receber as comunicações sobre esse caso.
12. Quando terminar, escolha Submit (Enviar).

Depois que recebermos sua solicitação, forneceremos uma resposta inicial em até 24 horas. Poderemos entrar em contato com você para solicitar informações adicionais. Se pudermos fornecer um código simplificado, enviaremos informações sobre os custos associados à obtenção de um código simplificado no país ou na região especificado na solicitação. Também forneceremos uma estimativa do tempo necessário para provisionar um código simplificado em seu país ou região. Em geral, leva entre algumas semanas para provisionar um código simplificado, embora esse atraso possa ser muito menor ou muito mais longo de acordo com o país ou a região em que o código simplificado esteja baseado.

 Note

As taxas associadas ao uso de códigos simplificados começam a ser cobradas imediatamente após o início de sua solicitação de códigos curtos com as operadoras. Você é responsável pelo pagamento dessas cobranças, mesmo se o código simplificado ainda não tiver sido totalmente provisionado.

Para evitar que nossos sistemas sejam usados para enviar conteúdo indesejado ou malicioso, consideramos cuidadosamente cada solicitação. Se o seu caso de uso não estiver alinhado com nossas políticas, talvez não seja possível atender à sua solicitação.

## Próximas etapas

Você registrou um código simplificado com as operadoras sem fio e analisou as configurações no console do Amazon SNS. Agora, pode usar o Amazon SNS para enviar mensagens SMS com seu código simplificado como número de origem.

## Solicitar números 10DLC, números de chamada gratuita e códigos longos P2P para mensagens SMS com o Amazon SNS

 Important

A partir de 1º de junho de 2021, os provedores de telecomunicações dos EUA não oferecem mais suporte ao uso de códigos longos person-to-person (P2P) para comunicações application-to-person (A2P) com destinos nos EUA. Em vez disso, você precisa usar outro tipo de ID de origem para essas mensagens. Para ter mais informações, consulte [10DLC](#).

Para solicitar [números 10DLC](#), [números de chamada gratuita](#) e [códigos longos P2P](#), use o console do Amazon Pinpoint. Para obter instruções detalhadas, consulte [Requesting a number](#) (“Solicitar um número”) no Manual do usuário do Amazon Pinpoint.

 Important

As operadoras de telefonia móvel dos EUA mudaram recentemente seus regulamentos e exigirão que todos os números de chamada gratuita (TFNs) concluam um processo de registro em um órgão regulador antes de 30 de setembro de 2022. Para obter mais

informações sobre o registro de um número de chamada gratuita, consulte [Registrar seu número de chamada gratuita](#).

Se você comprou seu número de chamada gratuita até 30 de setembro de 2022, seu status ficará ativo até 1.º de outubro de 2022, a menos que tenha concluído o registro e ele tenha retornado com o estado definido como concluído. Do contrário, ele será colocado no estado pendente e você não poderá enviar mensagens enquanto não registrar o número, o registro não for retornado ou o registro não for definido como ativo.

O registro pode levar até 15 dias úteis.

Para registrar sua empresa e campanha no 10DLC, abra o console Amazon Pinpoint na região Leste dos EUA (Norte da Virgínia). Em vez de solicitar um número de 10 DLC, use o [console de Quotas de AWS Serviço](#) para criar um caso de aumento do limite de serviço enquanto solicita o número de 10 DLC para essa região. Para obter informações sobre as regiões em que o Amazon Pinpoint está disponível, consulte [Endpoints e cotas do Amazon Pinpoint](#) na Referência geral da AWS.

#### Note

Se você for novo em relação a sistemas de mensagens por SMS com o Amazon SNS, solicite um limite mensal de gastos de SMS que atenda às demandas esperadas de seu caso de uso do SMS. Por padrão, o limite de gastos mensal é USD 1,00. Para ter mais informações, consulte [Solicitar aumentos de sua cota de gastos mensais de SMS para o Amazon SNS](#).

## Solicitar IDs do remetente para sistema de mensagens por SMS com o Amazon SNS

#### Important

Se você for novo em relação ao sistema de mensagens por SMS com o Amazon SNS, solicite um limite mensal de gastos de SMS que atenda às demandas de SMS esperadas de seu caso de uso. Por padrão, o limite de gastos mensal é 1,00 USD. Você pode solicitar o aumento do limite de gastos no mesmo caso de suporte que inclui sua solicitação de um ID do remetente. Ou, se preferir, pode abrir um caso. Para obter mais informações, consulte [Solicitar aumentos de sua cota de gastos mensais de SMS para o Amazon SNS](#).

Nas mensagens SMS, um ID do remetente é um nome exibido como o remetente da mensagem nos dispositivos dos destinatários. Os IDs do remetente são uma maneira útil de se identificar para os destinatários de suas mensagens.

O suporte a IDs de remetentes varia de acordo com o país . Por exemplo, as operadoras nos Estados Unidos não oferecem suporte a IDs de remetente, mas as operadoras na Índia exigem que os remetentes usem IDs de remetente. Para obter uma lista de países que oferecem suporte a IDs de remetente, consulte [Países e regiões compatíveis](#).

#### Important

Alguns países exigem que você registre IDs de remetente antes de usá-los para enviar mensagens. Dependendo do país, esse processo de registro pode levar várias semanas. Os países que exigem IDs de remetente pré-registrados são indicados na tabela na página [Países com suporte](#).

Você pode usar e registrar o mesmo ID de remetente em várias contas da AWS para mensagens SMS. Se você tem suporte corporativo e está registrando vários modelos em várias contas, siga as etapas abaixo e trabalhe com o gerente de conta técnico para garantir que sua experiência de integração seja coordenada.

Se você estiver enviando mensagens para destinatários em um país com suporte para os IDs de remetente, e esse país não exigir que você registre o ID de remetente, não será necessário executar nenhuma etapa adicional. É possível começar a enviar mensagens que incluem valores de ID de remetente imediatamente.

Basta concluir os procedimentos desta página se planeja enviar mensagens para um país onde o registro de IDs de remetente é necessário.

#### Etapa 1: abrir um caso de SMS do Amazon SNS

Se você planeja enviar mensagens a destinatários de um país em que os IDs de remetente são necessários, é possível solicitar um ID de remetente criando um novo caso no AWS Support Center.

#### Note

Se você planeja enviar mensagens para destinatários de um país em que os IDs de remetente são permitidos, mas não são necessários, não é necessário abrir um caso no

Support Center. É possível começar a enviar mensagens que usam IDs de remetente imediatamente.

Para solicitar um ID do remetente

1. Faça login no AWS Management Console em <https://console.aws.amazon.com/>.
2. No menu Support (Suporte), escolha Support Center.
3. No painel Seus casos de suporte, escolha Criar caso.
4. Escolha o link Procurando aumentos no limite de serviço? e preencha o seguinte:
  - Em Limit type (Tipo de limite), selecione Pinpoint SMS (Identificar SMS).
  - (Opcional) Em Fornecer um link ao site à aplicação que enviará mensagens SMS, identifique o site ou a aplicação em que os membros do público optam por receber suas mensagens SMS.
  - (Opcional) Em Qual tipo de mensagem você pretende enviar, escolha o tipo de mensagem que pretende enviar utilizando o código longo:
    - One Time Passwords (Senhas de uso único): mensagens que fornecem senhas que seus clientes usam para se autenticarem em seu site ou aplicação.
    - Promotional (Promocional): mensagens não críticas que promovem o seu negócio ou serviço, como ofertas especiais ou anúncios.
    - Transactional (Transacional): mensagens informativas importantes que oferecem suporte para transações do cliente, como confirmações de pedidos ou alertas de contas. As mensagens transacionais não devem conter conteúdo promocional nem de marketing.
  - (Opcional) Em De qual região da AWS você enviará mensagens, escolha a Região da AWS de onde você enviará mensagens SMS.
  - (Opcional) Em Para quais países você planeja enviar mensagens, insira os países onde você deseja registrar um ID de remetente. Os requisitos de registro e a compatibilidade com os IDs de remetente variam de acordo com o país. Para obter mais informações, consulte [Países e regiões compatíveis](#).

Se a lista de países exceder o número de caracteres permitido por essa caixa de texto, será possível listar os países na seção Descrição do caso.

- (Opcional) Em Como seus clientes optam por receber suas mensagens, forneça uma descrição de como seus clientes optam por receber suas mensagens.

- (Opcional) No campo Forneça o modelo de mensagem que você planeja usar para enviar mensagens aos seus clientes, inclua todos os modelos de mensagem que você usará.
5. Em Requests (Solicitações), preencha as seguintes seções:
- Em Região, escolha a Região da AWS para o ID de remetente.
-  **Note**

A região é obrigatória na seção Solicitações. Mesmo que você tenha fornecido essas informações na seção Detalhes do caso, também é necessário incluí-las aqui.
- Em Resource Type (Tipo de recurso), escolha General Limits (Limites gerais).
  - Em Limite, escolha Acesso à produção de SMS.
6. Em Novo valor de limite, selecione o número de IDs de remetente que você está solicitando. Normalmente, esse valor é **1**.
7. (Opcional) Se você quiser enviar outras solicitações, escolha Adicionar outra solicitação. Para obter as informações necessárias, consulte as outras seções em [Solicitar suporte para sistema de mensagens SMS com o Amazon SNS](#).
8. Em Case description (Descrição do caso), em Use case description (Descrição do caso de uso), forneça os seguintes detalhes:
- O ID de remetente que você deseja registrar.
  - O modelo do que você planeja usar para suas mensagens SMS.
  - O número de mensagens que você planeja enviar para cada destinatário por mês.
  - Informações sobre como seus clientes optam por receber suas mensagens.
  - O nome da sua empresa ou organização.
  - O endereço associado à sua empresa ou organização.
  - O país de sede de sua empresa ou organização.
  - Um número de telefone de sua empresa ou organização.
  - O URL do site de sua empresa ou organização.
9. Em Opções de contato, para Idioma de contato preferencial, escolha o idioma no qual você deseja receber as comunicações sobre esse caso.
10. Quando terminar, escolha Submit (Enviar).

Depois que recebermos sua solicitação, forneceremos uma resposta inicial em até 24 horas. Poderemos entrar em contato com você para solicitar informações adicionais. Se pudermos fornecer um ID do remetente, enviaremos uma estimativa do tempo necessário para provisioná-lo.

Para evitar que nossos sistemas sejam usados para enviar conteúdo indesejado ou malicioso, consideramos cuidadosamente cada solicitação. Se o seu caso de uso não estiver alinhado com nossas políticas, talvez não seja possível atender à sua solicitação.

## Etapa 2: atualizar suas configurações de SMS no console do Amazon SNS

Quando você concluir o processo de obtenção do ID do remetente, responderemos ao seu caso. Quando você receber esta notificação, execute as etapas nesta seção para configurar o Amazon SNS para usar o ID de remetente como padrão para todas as mensagens enviadas usando a sua conta. Como alternativa, você pode optar por especificar qual ID de remetente deve ser usado ao [publicar a mensagem](#).

1. Faça login no [console do Amazon SNS](#).
2. No painel de navegação, escolha Dispositivo móvel e Mensagens de texto (SMS).
3. Na seção Preferências de mensagens SMS, escolha Editar.
4. Na seção Detalhes, no campo ID padrão do remetente, insira o ID de remetente fornecido a ser usado como padrão para todas as mensagens da sua conta.
5. Quando terminar, escolha Salvar alterações.

## Próximas etapas

Você registrou um ID do remetente e atualizou as configurações no console do Amazon SNS. Agora, é possível usar o Amazon SNS para enviar mensagens SMS com o ID do remetente. Os destinatários de SMS em países compatíveis verão nos dispositivos o ID do remetente como o remetente da mensagem. Se for usado um ID de remetente diferente ao publicar as mensagens, ele substituirá o ID padrão configurado aqui.

## Solicitar aumentos de sua cota de gastos mensais de SMS para o Amazon SNS

O Amazon SNS fornece cotas de gastos para ajudar você a gerenciar o custo máximo por mês incorrido pelo envio de SMS usando sua conta. A cota de gastos limita seu risco em caso de ataque malicioso e impede que sua aplicação upstream envie mais mensagens do que o esperado. Você pode configurar o Amazon SNS para parar o envio de mensagens SMS quando ele determinar que esse envio tiver um custo que ultrapassará sua cota de gastos do mês.

Para garantir que suas operações não sejam afetadas, recomendamos solicitar uma cota de gastos alta o suficiente para suportar suas cargas de trabalho de produção. Para obter mais informações, consulte [Etapa 1: abrir um caso de SMS do Amazon SNS](#). Depois de receber a cota, você pode gerenciar seu risco aplicando a cota completa ou um valor menor, conforme descrito em [Etapa 2: atualizar suas configurações de SMS](#). Ao aplicar um valor mais baixo, você pode controlar seus gastos mensais com a opção de escalar, se necessário.

 Important

Como o Amazon SNS é um sistema distribuído, ele interromperá o envio de mensagens SMS minutos depois que a cota de gasto for excedida. Durante esse intervalo, se você continuar a enviar mensagens SMS, poderá gerar custos que ultrapassam sua cota.

Definimos a cota de gastos para todas as novas contas como 1,00 USD por mês. Essa cota tem como objetivo permitir testar os recursos de envio de mensagens do Amazon SNS. Para solicitar um aumento da cota de gastos de SMS para sua conta, abra um caso de aumento de cota de no AWS Support Center.

#### Etapa 1: abrir um caso de SMS do Amazon SNS

Você pode solicitar um aumento de sua cota de gastos mensais abrindo um caso de aumento de cota no AWS Support Center.

 Note

Alguns campos no formulário de solicitação são marcados como "opcionais". No entanto, o AWS Support exige todas as informações mencionadas nas etapas a seguir para processar sua solicitação. Se você não fornecer todas as informações necessárias, poderá haver atrasos no processamento de sua solicitação.

1. Faça login no AWS Management Console em <https://console.aws.amazon.com/>.
2. No menu Support (Suporte), escolha Support Center.
3. No painel Seus casos de suporte, escolha Criar caso.
4. Escolha o link Procurando aumentos no limite de serviço? e preencha o seguinte:
  - No Tipo de limite, escolha Mensagens de texto do SNS.

- (Opcional) Em Fornecer um link ao site ou à aplicação que enviará mensagens SMS, forneça informações sobre o site, a aplicação ou o serviço que enviará mensagens SMS.
  - (Opcional) Em Qual tipo de mensagem você pretende enviar, escolha o tipo de mensagem que pretende enviar utilizando o código longo:
    - One Time Passwords (Senhas de uso único): mensagens que fornecem senhas que seus clientes usam para se autenticarem em seu site ou aplicação.
    - Promotional (Promocional): mensagens não críticas que promovem o seu negócio ou serviço, como ofertas especiais ou anúncios.
    - Transactional (Transacional): mensagens informativas importantes que oferecem suporte para transações do cliente, como confirmações de pedidos ou alertas de contas. As mensagens transacionais não devem conter conteúdo promocional nem de marketing.
  - (Opcional) Em De qual região da AWS você enviará mensagens, escolha a região de onde você enviará mensagens.
  - (Opcional) Em Para quais países você planeja enviar mensagens, insira o país ou a região em que deseja comprar códigos simplificados.
  - (Opcional) Em Como seus clientes optam por receber suas mensagens, forneça detalhes sobre seu processo de consentimento.
  - (Opcional) No campo Forneça o modelo de mensagem que você planeja usar para enviar mensagens aos seus clientes, inclua o modelo que será usado.
5. Em Requests (Solicitações), preencha as seguintes seções:
- Para Região, escolha a região da qual você enviará mensagens.

 Note

A região é obrigatória na seção Solicitações. Mesmo que você tenha fornecido essas informações na seção Detalhes do caso, também é necessário incluí-las aqui.

- Em Resource Type (Tipo de recurso), escolha General Limits (Limites gerais).
  - Em Limit (Limite), escolha Account Spend Threshold Increase (Aumento de limite de gasto da conta).
6. Em “Novo valor de limite”, insira o valor máximo (em USD) que você pode gastar em SMS a cada mês.

7. Em Case description (Descrição do caso), em Use case description (Descrição do caso de uso), forneça os seguintes detalhes:
  - O site ou o aplicativo da empresa ou do serviço que está enviando mensagens SMS.
  - O serviço que é fornecido pelo seu site ou aplicativo e como suas mensagens SMS contribuem para esse serviço.
  - Como os usuários se cadastram para receber voluntariamente suas mensagens SMS no seu site, aplicativo ou em outro local.

Se a cota de gastos solicitada (o valor que você especificou em New quota value (Novo valor de cota)) exceder 10.000 USD, forneça os detalhes adicionais a seguir para cada país de destino do sistema de mensagens:

  - Se você estiver usando um ID do remetente ou código simplificado. Se você está usando um ID do remetente, forneça:
    - O ID do remetente.
    - Se o ID do remetente está registrado com operadoras sem fio no país.
  - O máximo esperado de transações por segundo (TPS) para suas mensagens.
  - O tamanho médio das mensagens.
  - O modelo para as mensagens que você envia para o país.
  - (Opcional) Necessidades de codificação de caracteres, se houver.

8. (Opcional) Se você quiser enviar outras solicitações, escolha Adicionar outra solicitação. Se você incluir várias solicitações, forneça as informações necessárias para cada uma delas. Para obter as informações necessárias, consulte as outras seções em [Solicitar suporte para sistema de mensagens SMS com o Amazon SNS](#).
9. Em Opções de contato, para Idioma de contato preferencial, escolha o idioma no qual você deseja receber as comunicações sobre esse caso.
10. Quando terminar, escolha Submit (Enviar).

A equipe do AWS Support fornece uma resposta inicial para sua solicitação em 24 horas.

Para evitar que nossos sistemas sejam usados para enviar conteúdo indesejado ou malicioso, consideramos cuidadosamente cada solicitação. Se for possível, atenderemos à sua solicitação dentro desse período de 24 horas. No entanto, se for necessário solicitar mais informações a você, o tempo de resolução poderá ser mais longo.

Se o seu caso de uso não estiver alinhado com nossas políticas, talvez não seja possível atender à sua solicitação.

## Etapa 2: atualizar suas configurações de SMS no console do Amazon SNS

Depois de receber uma notificação de que a cota de gastos mensais foi aumentada, ajuste a cota de gastos de sua conta no console do Amazon SNS.

### Important

Você deve concluir as etapas a seguir para que seu limite de gastos de SMS seja aumentado.

Como ajustar sua cota de gastos no console

1. Faça login no [console do Amazon SNS](#).
2. Abra o menu de navegação à esquerda, expanda Dispositivos móveis e escolha Mensagens de texto (SMS).
3. Na página Mobile text messaging (SMS) [Mensagens de texto (SMS) em dispositivos móveis], na seção Text messaging preferences (Preferências de mensagens de texto), escolha Edit (Editar).
4. Na página Editar preferências de mensagens de texto, na seção Detalhes, insira o novo limite de gastos de SMS no campo Limite de gastos da conta.

### Note

É possível receber um aviso de que o valor inserido é maior que o limite de gastos padrão. Pode ignorar esse erro.

5. Escolha Save changes (Salvar alterações).

### Note

Se receber um erro “Parâmetro inválido”, verifique o contato do AWS Support e confirme se o novo limite de gastos de SMS foi inserido corretamente. Se você ainda tiver algum problema, abra um caso na AWS Support Center.

## Definir preferências de mensagens SMS

Use o Amazon SNS para especificar preferências para mensagens SMS. Por exemplo, é possível especificar se as entregas serão otimizadas para fins de custo ou confiabilidade, o limite de gastos mensais, como as entregas serão registradas e a inscrição em relatórios diários de uso de SMS.

Essas preferências entrarão em vigor para cada mensagem SMS que você enviar de sua conta, mas será possível substituir algumas delas quando enviar uma mensagem individual. Para ter mais informações, consulte [Publicar em um telefone celular](#).

### Tópicos

- [Configurando as preferências de mensagens SMS usando o AWS Management Console](#)
- [Preferências de configuração \(AWS SDKs\)](#)

## Configurando as preferências de mensagens SMS usando o AWS Management Console

1. Faça login no [console do Amazon SNS](#).
2. Escolha uma [região que ofereça suporte a mensagens SMS](#).
3. No painel de navegação, escolha Mobile (Dispositivo Móvel), Text messaging (SMS) (Mensagens de texto (SMS)).
4. Na página Mobile text messaging (SMS) [Mensagens de texto (SMS) em dispositivos móveis], na seção Text messaging preferences (Preferências de mensagens de texto), escolha Edit (Editar).
5. Na página Editar preferências de mensagens SMS, na seção Detalhes, faça o seguinte:
  - a. Em Default message type (Tipo de mensagem padrão), selecione uma das seguintes opções:
    - Promocional: mensagens não essenciais (por exemplo, de marketing). O Amazon SNS otimiza a entrega de mensagens para gerar o custo mais baixo.
    - Transnacional (padrão): mensagens urgentes que comportam transações do cliente, como senhas únicas para autenticação multifator. O Amazon SNS otimiza a entrega de mensagens para gerar a mais alta confiabilidade.

Para obter informações sobre a definição de preços para mensagens promocionais e transacionais, consulte [Definição global de preço para SMS](#).

- b. (Opcional) Em Account spend limit (Limite de gastos da conta), insira a quantidade máxima (em dólares americanos) que você deseja gastar em mensagens SMS a cada mês.

 Important

- Por padrão, a cota de gasto é definida como 1,00 USD. Se desejar aumentar a cota de serviço, [envie uma solicitação](#).
- Se o valor definido no console exceder sua cota de serviço, o Amazon SNS interromperá a publicação de mensagens SMS.
- Como o Amazon SNS é um sistema distribuído, ele interromperá o envio de mensagens SMS minutos depois que a cota de gasto for excedida. Durante esse intervalo, se você continuar a enviar mensagens SMS, poderá incorrer em custos que excederão sua cota.

6. (Opcional) Em Default sender ID (ID do remetente padrão), insira um ID personalizado, como a marca de sua empresa, que será exibido como o remetente do dispositivo receptor.

 Note

O suporte a IDs de remetente varia de acordo com o país.

7. (Opcional) Digite o Amazon S3 bucket name for usage reports (Nome do bucket do Amazon S3 para relatórios de uso).

 Note

A política de bucket do S3 deve conceder acesso de gravação ao Amazon SNS.

8. Escolha Salvar alterações.

## Preferências de configuração (AWS SDKs)

Para definir suas preferências de SMS usando um dos AWS SDKs, use a ação nesse SDK que corresponde à `SetSMSAttributes` solicitação na API do Amazon SNS. Com essa solicitação, você atribui valores para os diferentes atributos de SMS, como sua cota de gasto mensal e seu tipo de SMS padrão (promocional ou transacional). Para todos os atributos de SMS, consulte [SetSMSAttributes](#) na Referência de API do Amazon Simple Notification Service.

Os exemplos de código a seguir mostram como usar `SetSMSAttributes`.

### C++

#### SDK para C++

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

Como usar o Amazon SNS para definir o atributo `DefaultSMSType`.

```
#!/ Set the default settings for sending SMS messages.
/*!
  \param smsType: The type of SMS message that you will send by default.
  \param clientConfiguration: AWS client configuration.
  \return bool: Function succeeded.
*/
bool AwsDoc::SNS::setSMSType(const Aws::String &smsType,
                             const Aws::Client::ClientConfiguration
&clientConfiguration) {
    Aws::SNS::SNSClient snsClient(clientConfiguration);

    Aws::SNS::Model::SetSMSAttributesRequest request;
    request.AddAttributes("DefaultSMSType", smsType);

    const Aws::SNS::Model::SetSMSAttributesOutcome outcome =
snsClient.SetSMSAttributes(
    request);

    if (outcome.IsSuccess()) {
```

```
        std::cout << "SMS Type set successfully " << std::endl;
    }
    else {
        std::cerr << "Error while setting SMS Type: '"
            << outcome.GetError().GetMessage()
            << "'" << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Para receber detalhes da API, consulte [SetSMSAttributes](#) na Referência da API AWS SDK for C++ .

## CLI

### AWS CLI

Para definir atributos de mensagens SMS

O exemplo `set-sms-attributes` a seguir define o ID do remetente padrão para mensagens SMS como `MyName`.

```
aws sns set-sms-attributes \
    --attributes DefaultSenderId=MyName
```

Este comando não produz saída.

- Para encontrar detalhes da API, consulte [SetSMSAttributes](#) na Referência de comandos da AWS CLI .

## Java

### SDK para Java 2.x

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.sns.SnsClient;
import software.amazon.awssdk.services.sns.model.SetSmsAttributesRequest;
import software.amazon.awssdk.services.sns.model.SetSmsAttributesResponse;
import software.amazon.awssdk.services.sns.model.SnsException;
import java.util.HashMap;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */
public class SetSMSAttributes {
    public static void main(String[] args) {
        HashMap<String, String> attributes = new HashMap<>(1);
        attributes.put("DefaultSMSType", "Transactional");
        attributes.put("UsageReportS3Bucket", "janbucket");

        SnsClient snsClient = SnsClient.builder()
            .region(Region.US_EAST_1)
            .build();
        setSMSAttributes(snsClient, attributes);
        snsClient.close();
    }

    public static void setSMSAttributes(SnsClient snsClient, HashMap<String,
String> attributes) {
        try {
            SetSmsAttributesRequest request = SetSmsAttributesRequest.builder()
                .attributes(attributes)
                .build();

            SetSmsAttributesResponse result =
snsClient.setSMSAttributes(request);
            System.out.println("Set default Attributes to " + attributes + ".
Status was "
                + result.sdkHttpResponse().statusCode());
        } catch (SnsException e) {
```

```
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}
```

- Para receber detalhes da API, consulte [SetSMSAttributes](#) na Referência da API AWS SDK for Java 2.x .

## JavaScript

### SDK para JavaScript (v3)

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

Crie o cliente em um módulo separado e exporte-o.

```
import { SNSClient } from "@aws-sdk/client-sns";

// The AWS Region can be provided here using the `region` property. If you leave
it blank
// the SDK will default to the region set in your AWS config.
export const snsClient = new SNSClient({});
```

Importe o SDK e os módulos do cliente e chame a API.

```
import { SetSMSAttributesCommand } from "@aws-sdk/client-sns";
import { snsClient } from "../libs/snsClient.js";

/**
 * @param {"Transactional" | "Promotional"} defaultSmsType
 */
export const setSmsType = async (defaultSmsType = "Transactional") => {
  const response = await snsClient.send(
    new SetSMSAttributesCommand({
```

```
        attributes: {
            // Promotional - (Default) Noncritical messages, such as marketing
            // messages.
            // Transactional - Critical messages that support customer transactions,
            // such as one-time passcodes for multi-factor authentication.
            DefaultSMSType: defaultSmsType,
        },
    })),
);
console.log(response);
// {
//   '$metadata': {
//     httpStatusCode: 200,
//     requestId: '1885b977-2d7e-535e-8214-e44be727e265',
//     extendedRequestId: undefined,
//     cfId: undefined,
//     attempts: 1,
//     totalRetryDelay: 0
//   }
// }
return response;
};
```

- Para obter mais informações, consulte o [Guia do desenvolvedor do AWS SDK for JavaScript](#).
- Para receber detalhes da API, consulte [SetSMSAttributes](#) na Referência da API AWS SDK for JavaScript .

## PHP

### SDK para PHP

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
$SnSclient = new SnsClient([
    'profile' => 'default',
```

```
'region' => 'us-east-1',
'version' => '2010-03-31'
]);

try {
    $result = $SnSClient->SetSMSAttributes([
        'attributes' => [
            'DefaultSMSType' => 'Transactional',
        ],
    ]);
    var_dump($result);
} catch (AwsException $e) {
    // output error message if fails
    error_log($e->getMessage());
}
```

- Para obter mais informações, consulte o [Guia do desenvolvedor do AWS SDK for PHP](#).
- Para receber detalhes da API, consulte [SetSMSAttributes](#) na Referência da API do AWS SDK for PHP .

## Enviar mensagens SMS

Esta seção descreve como enviar mensagens SMS.

### Tópicos

- [Publicar em um tópico](#)
- [Publicar em um telefone celular](#)

### Publicar em um tópico

É possível publicar uma única mensagem SMS em vários números de telefone de uma só vez inscrevendo esses números de telefone em um tópico do Amazon SNS. Um tópico do Amazon SNS é um canal de comunicação para o qual é possível adicionar inscritos e publicar mensagens para todos esses inscritos. Um inscrito receberá todas as mensagens publicadas no tópico até você cancelar a inscrição ou ele cancelar o recebimento de mensagens SMS de sua conta da AWS.

### Tópicos

- [Enviar uma mensagem para um tópico \(console\)](#)
- [Enviar uma mensagem para um tópico \(AWS SDKs\)](#)

## Enviar uma mensagem para um tópico (console)

### Para criar um tópico

Realize as seguintes etapas se você ainda não tiver um tópico para o qual deseja enviar mensagens SMS.

1. Faça login no [console do Amazon SNS](#).
2. No menu do console, selecione uma [Região da AWS compatível com mensagens SMS](#).
3. No painel de navegação, escolha Topics (Tópicos).
4. Na página Topics (Tópicos), escolha Create topic (Criar tópico).
5. Na página Create topic (Criar tópico), em Details (Detalhes), faça o seguinte:
  - a. Em Tipo, escolha Padrão.
  - b. Em Name (Nome), insira um nome para o tópico.
  - c. (Opcional) Para Display name (Nome de exibição), digite um prefixo personalizado para suas mensagens SMS. Quando você envia uma mensagem para o tópico, o Amazon SNS acrescenta o nome de exibição seguido por uma seta para a direita (>) e um espaço. Os nomes de exibição não fazem distinção entre maiúsculas e minúsculas e o Amazon SNS converte os nomes de exibição para caracteres maiúsculos. Por exemplo, se o nome de exibição de um tópico é MyTopic e a mensagem é Hello World!, a mensagem é exibida como:

```
MYTOPIC> Hello World!
```
6. Escolha Criar tópico. O nome do tópico e o nome de recurso da Amazon (ARN) aparecem na página Topics (Tópicos).

### Para criar uma assinatura de SMS

Use assinaturas para enviar uma mensagem SMS para vários destinatários publicando a mensagem somente uma vez em seu tópico.

**Note**

Ao começar a usar o Amazon SNS para enviar mensagens SMS, a conta da AWS estará na sandbox do SMS. A sandbox de SMS oferece um ambiente seguro para testar os recursos do Amazon SNS sem arriscar a reputação como remetente do SMS. Enquanto sua conta estiver na sandbox de SMS, é possível usar todos os recursos do Amazon SNS, mas as mensagens SMS podem ser enviadas somente para números de telefone de destino verificados. Para obter mais informações, consulte [Sandbox de SMS](#).

1. Faça login no [console do Amazon SNS](#).
2. No painel de navegação, escolha Subscriptions.
3. Na página Subscriptions (Assinaturas), escolha Create subscription (Criar assinatura).
4. Na página Create subscription (Criar assinatura), na seção Details (Detalhes), faça o seguinte:
  - a. Em Topic ARN (ARN do tópico), insira ou escolha o nome do recurso da Amazon (ARN) do tópico para o qual você deseja enviar mensagens SMS.
  - b. Para Protocol (Protocolo), escolha SMS.
  - c. Para Endpoint digite o número de telefone que você deseja assinar para seu tópico.
5. Selecione Criar assinatura. As informações da assinatura são exibidas na página Subscriptions (Assinaturas).

Para adicionar mais números de telefone, repita essas etapas. Também é possível adicionar outros tipos de assinaturas, como e-mail.

### Para enviar uma mensagem

Quando você publica uma mensagem em um tópico, o Amazon SNS tenta entregar essa mensagem para cada número de telefone inscrito no tópico.

1. No [console do Amazon SNS](#), na página Topics (Tópicos), escolha o nome do tópico para o qual você deseja enviar mensagens SMS.
2. Na página de detalhes do tópico, selecione Publicar mensagem.
3. Na página Publish message to topic (Publicar mensagem no tópico), em Message details (Detalhes da mensagem), faça o seguinte:

- a. Em Subject (Assunto), deixe o campo em branco, a menos que o tópico contenha inscrições de e-mail e você queira publicar nas inscrições por e-mail e por SMS. O Amazon SNS usará o Subject (Assunto) que você inserir como linha de assunto do e-mail.
  - b. (Opcional) Para Time to Live (TTL) (Vida útil (TTL)), insira o número de segundos que o Amazon SNS tem para enviar sua mensagem SMS para qualquer assinante de endpoint de aplicação móvel.
4. Em Message body (Corpo da mensagem), faça o seguinte:
- a. Em Message structure (Estrutura de mensagem), escolha Identical payload for all delivery protocols (Carga útil idêntica para todos os protocolos de entrega) para enviar a mesma mensagem para todos os tipos de protocolo inscritos no tópico. Ou escolha Custom payload for each delivery protocol (Carga útil personalizada para cada protocolo de entrega) para personalizar a mensagem direcionada a assinantes de diferentes tipos de protocolo. Por exemplo, é possível inserir uma mensagem padrão para assinantes de número de telefone e uma mensagem personalizada para assinantes de e-mail.
  - b. Em Message body to send to the endpoint (Corpo da mensagem a ser enviada para o endpoint), insira sua mensagem ou suas mensagens personalizadas por protocolo de entrega.

Se o tópico tem um nome de exibição, o Amazon SNS adiciona-o à mensagem, o que aumenta o tamanho da mensagem. O tamanho do nome de exibição é o número de caracteres no nome, mais dois caracteres para a seta (>) e o espaço adicionado pelo Amazon SNS.

Para obter mais informações sobre as cotas de tamanho de mensagens SMS, consulte [Publicar em um telefone celular](#).

5. (Opcional) Em Message attributes (Atributos de mensagem), adicione os metadados das mensagens, como carimbos de data/hora, assinaturas e IDs.
6. Selecione Publish message (Publicar mensagem). O Amazon SNS envia a mensagem SMS e exibe uma mensagem de êxito.

## Enviar uma mensagem para um tópico (AWS SDKs)

Para usar um AWS SDK, configure-o com suas credenciais. Para obter mais informações, consulte [The shared config and credentials files](#) (“Os arquivos compartilhados de configuração e de credenciais no Guia de referência de AWS SDKs e ferramentas).

O exemplo de código a seguir mostra como:

- Crie um tópico do Amazon SNS.
- Inscrever números de telefone no tópico.
- Publicar mensagens SMS no tópico para que todos os números de telefone inscritos recebam a mensagem de uma só vez.

## Java

### SDK para Java 2.x

#### Note

Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

Criar um tópico e retorne seu ARN.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.sns.SnsClient;
import software.amazon.awssdk.services.sns.model.CreateTopicRequest;
import software.amazon.awssdk.services.sns.model.CreateTopicResponse;
import software.amazon.awssdk.services.sns.model.SnsException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
 * started.html
 */
public class CreateTopic {
    public static void main(String[] args) {
        final String usage = ""

                Usage:    <topicName>

                Where:
```

```
        topicName - The name of the topic to create (for example,
mytopic).

        """;

    if (args.length != 1) {
        System.out.println(usage);
        System.exit(1);
    }

    String topicName = args[0];
    System.out.println("Creating a topic with name: " + topicName);
    SnsClient snsClient = SnsClient.builder()
        .region(Region.US_EAST_1)
        .build();

    String arnVal = createSNSTopic(snsClient, topicName);
    System.out.println("The topic ARN is" + arnVal);
    snsClient.close();
}

public static String createSNSTopic(SnsClient snsClient, String topicName) {
    CreateTopicResponse result;
    try {
        CreateTopicRequest request = CreateTopicRequest.builder()
            .name(topicName)
            .build();

        result = snsClient.createTopic(request);
        return result.topicArn();

    } catch (SnsException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
    return "";
}
}
```

Inscreva um endpoint em um tópico.

```
import software.amazon.awssdk.regions.Region;
```

```
import software.amazon.awssdk.services.sns.SnsClient;
import software.amazon.awssdk.services.sns.model.SnsException;
import software.amazon.awssdk.services.sns.model.SubscribeRequest;
import software.amazon.awssdk.services.sns.model.SubscribeResponse;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */
public class SubscribeTextSMS {
    public static void main(String[] args) {
        final String usage = ""

                Usage:    <topicArn> <phoneNumber>

                Where:
                    topicArn - The ARN of the topic to subscribe.
                    phoneNumber - A mobile phone number that receives
notifications (for example, +1XXX5550100).
                """;

        if (args.length < 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String topicArn = args[0];
        String phoneNumber = args[1];
        SnsClient snsClient = SnsClient.builder()
            .region(Region.US_EAST_1)
            .build();

        subTextSNS(snsClient, topicArn, phoneNumber);
        snsClient.close();
    }

    public static void subTextSNS(SnsClient snsClient, String topicArn, String
phoneNumber) {
        try {
```

```

        SubscribeRequest request = SubscribeRequest.builder()
            .protocol("sms")
            .endpoint(phoneNumber)
            .returnSubscriptionArn(true)
            .topicArn(topicArn)
            .build();

        SubscribeResponse result = snsClient.subscribe(request);
        System.out.println("Subscription ARN: " + result.subscriptionArn() +
"\n\n Status is "
            + result.sdkHttpResponse().statusCode());

    } catch (SnsException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}
}
}

```

Defina atributos na mensagem, como o ID do remetente, o preço máximo e seu tipo. Os atributos de mensagem são opcionais.

```

import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.sns.SnsClient;
import software.amazon.awssdk.services.sns.model.SetSmsAttributesRequest;
import software.amazon.awssdk.services.sns.model.SetSmsAttributesResponse;
import software.amazon.awssdk.services.sns.model.SnsException;
import java.util.HashMap;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */
public class SetSMSAttributes {
    public static void main(String[] args) {
        HashMap<String, String> attributes = new HashMap<>(1);
        attributes.put("DefaultSMSType", "Transactional");
    }
}

```

```
        attributes.put("UsageReportS3Bucket", "janbucket");

        SnsClient snsClient = SnsClient.builder()
            .region(Region.US_EAST_1)
            .build();
        setSNSAttributes(snsClient, attributes);
        snsClient.close();
    }

    public static void setSNSAttributes(SnsClient snsClient, HashMap<String,
String> attributes) {
        try {
            SetSmsAttributesRequest request = SetSmsAttributesRequest.builder()
                .attributes(attributes)
                .build();

            SetSmsAttributesResponse result =
snsClient.setSMSAttributes(request);
            System.out.println("Set default Attributes to " + attributes + ".
Status was "
                + result.sdkHttpResponse().statusCode());

        } catch (SnsException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

Publique uma mensagem em um tópico. A mensagem é enviada para todos os assinantes.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.sns.SnsClient;
import software.amazon.awssdk.services.sns.model.PublishRequest;
import software.amazon.awssdk.services.sns.model.PublishResponse;
import software.amazon.awssdk.services.sns.model.SnsException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
```

```
*
* https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
*/
public class PublishTextSMS {
    public static void main(String[] args) {
        final String usage = ""

            Usage:    <message> <phoneNumber>

            Where:
                message - The message text to send.
                phoneNumber - The mobile phone number to which a message is
sent (for example, +1XXX5550100).\s
            """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String message = args[0];
        String phoneNumber = args[1];
        SnsClient snsClient = SnsClient.builder()
            .region(Region.US_EAST_1)
            .build();
        pubTextSMS(snsClient, message, phoneNumber);
        snsClient.close();
    }

    public static void pubTextSMS(SnsClient snsClient, String message, String
phoneNumber) {
        try {
            PublishRequest request = PublishRequest.builder()
                .message(message)
                .phoneNumber(phoneNumber)
                .build();

            PublishResponse result = snsClient.publish(request);
            System.out
                .println(result.messageId() + " Message sent. Status was " +
result.sdkHttpResponse().statusCode());

        } catch (SnsException e) {
```

```
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}
```

## Publicar em um telefone celular

É possível usar o Amazon SNS para enviar mensagens SMS diretamente para um telefone celular sem inscrever o número de telefone em um tópico do Amazon SNS.

### Note

A inscrição de números de telefone em um tópico é útil se você quer enviar uma mensagem para vários números de telefone de uma só vez. Para obter instruções sobre como publicar uma mensagem SMS em um tópico, consulte [Publicar em um tópico](#).

Ao enviar uma mensagem, é possível controlar se a mensagem é otimizada para custos ou confiabilidade de entrega. Também é possível especificar um [ID do remetente ou número de origem](#). Se você enviar a mensagem programaticamente usando a API do Amazon SNS ou AWS os SDKs, poderá especificar um preço máximo para a entrega da mensagem.

Cada mensagem SMS pode ter até 140 bytes, e a cota de caracteres depende do esquema de codificação. Por exemplo, uma mensagem SMS pode incluir:

- 160 caracteres GSM
- 140 caracteres ASCII
- 70 caracteres UCS-2

Se você publicar uma mensagem que exceda a cota de tamanho, o Amazon SNS a enviará como várias mensagens, cada uma delas respeitando a cota de tamanho. Para isso, as palavras são mantidas inteiras na mensagem, não cortadas. O tamanho total da cota de uma única ação de publicação de SMS é de 1.600 bytes.

Ao enviar uma mensagem SMS, você especifica o número de telefone com o formato E.164, que é uma estrutura padrão de numeração de telefones para telecomunicação internacional. Os números

de telefone que seguem esse formato podem conter no máximo 15 dígitos junto como prefixo de um sinal de adição (+) e o código do país. Por exemplo, um número de telefone dos EUA no formato E.164 seria exibido como +1XXX5550100.

## Tópicos

- [Enviar uma mensagem \(console\)](#)
- [Enviando uma mensagem \(AWS SDKs\)](#)

### Enviar uma mensagem (console)

1. Faça login no [console do Amazon SNS](#).
2. No menu do console, selecione uma [Região da AWS compatível com mensagens SMS](#).
3. No painel de navegação, escolha Mensagens de texto (SMS).
4. Na página Mobile text messaging (SMS) (Mensagens de texto para dispositivos móveis (SMS)), selecione Publish text message (Publicar mensagem de texto).
5. Na página Publish SMS message (Publicar mensagem SMS), para Message type (Tipo de mensagem), escolha uma das seguintes opções:
  - Promotional (Promocional): mensagens não essenciais, como mensagens de marketing.
  - Transactional (Transacional): mensagens urgentes que oferecem suporte para transações do cliente, como senhas únicas para autenticação multifator.

#### Note

Essa configuração em nível de mensagem substitui o tipo de mensagem padrão em nível de conta. É possível definir um tipo de mensagem padrão em nível de conta na seção Text messaging preferences (Opções de mensagens de texto) da página Mobile text messaging (SMS) (Mensagens de texto para dispositivos móveis (SMS)).

Para obter informações sobre mensagens promocionais e transacionais, consulte [Worldwide SMS Pricing](#) (Preço global para SMS).

6. Em Destination phone number (Número de telefone de destino), digite o número de telefone para o qual você deseja enviar a mensagem.
7. Em Message (Mensagem), digite a mensagem a ser enviada.

8. (Opcional) Em Origination identities (Identidades de origem), especifique como você se identificará para seus destinatários:

- Para especificar um Sender ID (ID do remetente), digite um ID personalizado que contenha de 3 a 11 caracteres alfanuméricos sem espaços, com pelo menos uma letra. O ID do remetente é exibido como o remetente da mensagem no dispositivo receptor. Por exemplo, é possível usar a marca de sua empresa para tornar a origem da mensagem mais fácil de reconhecer.

O suporte a IDs de remetentes varia de acordo com o país e/ou a região. Por exemplo, as mensagens enviadas para os números de telefone dos EUA não exibirão o ID do remetente. Para saber quais países e regiões oferecem suporte a IDs de remetentes, consulte [Países e regiões compatíveis](#).

Se você não especificar um ID do remetente, uma das seguintes informações será exibida como identidade de origem:

- Em países com suporte para códigos longos, o código longo será exibido.
- Em países onde há suporte apenas para IDs do remetente, será exibido um AVISO.

Esse ID do remetente no nível de mensagem substitui o ID de remetente padrão, que você define na página Preferências de mensagens de texto.

- Para especificar um Origination number (Número de origem), insira uma string de 5 a 14 números para ser exibida como número de telefone do remetente no dispositivo do receptor. Essa cadeia de caracteres deve corresponder a um número de origem configurado em sua Conta da AWS para o país de destino. O número de origem pode ser um número 10DLC, um número gratuito, um código person-to-person longo ou um código curto. Para ter mais informações, consulte [Identidades de origem para mensagens SMS](#).

Se você não especificar um número de origem, o Amazon SNS selecionará um número de origem para usar na mensagem de texto SMS, com base na sua configuração. Conta da AWS

9. Se você estiver enviando mensagens SMS para destinatários na Índia, expanda Country-specific attributes (Atributos específicos do país) e especifique estes atributos:

- Entity ID (ID da entidade): o ID da entidade ou ID da Principal Entity (PE – Entidade principal) para enviar mensagens SMS a destinatários na Índia. Esse ID é uma string exclusiva de 1 a 50 caracteres que a Telecom Regulatory Authority of India (TRAI) fornece para identificar a entidade que você registrou junto a ela.

- **Template ID (ID do modelo):** o ID de modelo para enviar mensagens SMS a destinatários na Índia. Esse ID é uma string exclusiva de 1 a 50 caracteres fornecida pela TRAI que identifica o modelo que você registrou junto a ela. O ID do modelo deve ser associado ao ID do remetente que você especificou para a mensagem.

Para obter mais informações sobre como enviar mensagens SMS para destinatários na Índia, consulte [Requisitos para registro de ID de remetente da Índia](#).

10. Selecione Publish message (Publicar mensagem).

#### Tip

Para enviar mensagens SMS de um número de origem, você também pode escolher Origination numbers (Números de origem) no painel de navegação do console do Amazon SNS. Escolha um número de origem que inclua SMS na coluna Capacities (Recursos) e, em seguida, escolha Publish text message (Publicar mensagem de texto).

### Enviando uma mensagem (AWS SDKs)

Para enviar uma mensagem SMS usando um dos AWS SDKs, use a operação de API nesse SDK que corresponde à Publish solicitação na API do Amazon SNS. Com essa solicitação, você pode enviar uma mensagem SMS diretamente para um número de telefone. Você também pode usar o parâmetro `MessageAttributes` para definir valores para os seguintes nomes de atributos:

`AWS.SNS.SMS.SenderID`

Um ID personalizado que contenha de 3 a 11 caracteres alfanuméricos ou caracteres de hífen (-) sem espaços, com pelo menos uma letra. O ID do remetente é exibido como o remetente da mensagem no dispositivo receptor. Por exemplo, é possível usar a marca de sua empresa para ajudar a tornar a origem da mensagem mais fácil de reconhecer.

O suporte a IDs de remetentes varia de acordo com o país ou a região. Por exemplo, as mensagens enviadas para os números de telefone dos EUA não exibem o ID do remetente. Para obter uma lista dos países ou regiões que oferecem suporte a IDs do remetente, consulte [Países e regiões compatíveis](#).

Se você não especificar um ID do remetente, um [código longo](#) será exibido como o ID do remetente nos países ou regiões compatíveis. Para países ou regiões que exigem um ID do remetente alfabético, um AVISO será exibido como o ID do remetente.

Esse atributo de mensagem substitui o atributo no nível da conta `DefaultSenderId` que você pode definir usando a solicitação `SetSMSAttributes`.

#### `AWS.MM.SMS.OriginNumber`

Uma string personalizada de 5 a 14 números, que pode incluir um sinal de adição opcional à esquerda (+). Essa string numérica aparece como o número de telefone do remetente no dispositivo receptor. A sequência de caracteres deve corresponder a um número de origem configurado na sua AWS conta para o país de destino. O número de origem pode ser um número 10DLC, número gratuito, código longo person-to-person (P2P) ou código curto. Para ter mais informações, consulte [Números de origem](#).

Se você não especificar um número de originação, o Amazon SNS escolherá um número de origem com base na configuração da sua conta. AWS

#### `AWS.SNS.SMS.MaxPrice`

O preço máximo em USD que você está disposto a gastar para enviar a mensagem SMS. Se o Amazon SNS determinar que enviar a mensagem pode gerar um custo que ultrapassaria o preço máximo, ele não enviará a mensagem.

Esse atributo não tem efeito se seus custos de month-to-date SMS já tiverem excedido a cota definida para o `MonthlySpendLimit` atributo. É possível definir o atributo `MonthlySpendLimit` usando a solicitação `SetSMSAttributes`.

Se você está enviando a mensagem para um tópico do Amazon SNS, o preço máximo se aplica a cada entrega de mensagem para cada número de telefone inscrito no tópico.

#### `AWS.SNS.SMS.SMSType`

O tipo de mensagem que você está enviando:

- `Promotional` (padrão): mensagens não essenciais, como mensagens de marketing.
- `Transactional`: mensagens essenciais que oferecem suporte a transações do cliente, como senhas únicas para autenticação multifator.

Esse atributo de mensagem substitui o atributo no nível da conta `DefaultSMSType` que você pode definir usando a solicitação `SetSMSAttributes`.

## AWS.MM.SMS.EntityId

Esse atributo é necessário somente para enviar mensagens SMS a destinatários na Índia.

Esse é o ID da entidade ou ID da Principal Entity (PE – Entidade principal) para enviar mensagens SMS a destinatários na Índia. Esse ID é uma string exclusiva de 1 a 50 caracteres que a Telecom Regulatory Authority of India (TRAI) fornece para identificar a entidade que você registrou junto a ela.

## AWS.MM.SMS.TemplateId

Esse atributo é necessário somente para enviar mensagens SMS a destinatários na Índia.

Este é o seu modelo para enviar mensagens SMS a destinatários na Índia. Esse ID é uma string exclusiva de 1 a 50 caracteres fornecida pela TRAI que identifica o modelo que você registrou junto a ela. O ID do modelo deve ser associado ao ID do remetente que você especificou para a mensagem.

## Enviar uma mensagem

Os exemplos de código a seguir mostram como publicar mensagens SMS usando o Amazon SNS.

### .NET

#### AWS SDK for .NET

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```
namespace SNSMessageExample
{
    using System;
    using System.Threading.Tasks;
    using Amazon;
    using Amazon.SimpleNotificationService;
    using Amazon.SimpleNotificationService.Model;

    public class SNSMessage
    {
```

```
private AmazonSimpleNotificationServiceClient snsClient;

/// <summary>
/// Initializes a new instance of the <see cref="SNSMessage"/> class.
/// Constructs a new SNSMessage object initializing the Amazon Simple
/// Notification Service (Amazon SNS) client using the supplied
/// Region endpoint.
/// </summary>
/// <param name="regionEndpoint">The Amazon Region endpoint to use in
/// sending test messages with this object.</param>
public SNSMessage(RegionEndpoint regionEndpoint)
{
    snsClient = new
AmazonSimpleNotificationServiceClient(regionEndpoint);
}

/// <summary>
/// Sends the SMS message passed in the text parameter to the phone
number
/// in phoneNum.
/// </summary>
/// <param name="phoneNum">The ten-digit phone number to which the text
/// message will be sent.</param>
/// <param name="text">The text of the message to send.</param>
/// <returns>Async task.</returns>
public async Task SendTextMessageAsync(string phoneNum, string text)
{
    if (string.IsNullOrEmpty(phoneNum) || string.IsNullOrEmpty(text))
    {
        return;
    }

    // Now actually send the message.
    var request = new PublishRequest
    {
        Message = text,
        PhoneNumber = phoneNum,
    };

    try
    {
        var response = await snsClient.PublishAsync(request);
    }
    catch (Exception ex)
```

```
        {
            Console.WriteLine($"Error sending message: {ex}");
        }
    }
}
```

- Para obter detalhes da API, consulte [Publish](#) na Referência da API do AWS SDK for .NET .

## C++

### SDK for C++

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```
/**
 * Publish SMS: use Amazon Simple Notification Service (Amazon SNS) to send an
 * SMS text message to a phone number.
 * Note: This requires additional AWS configuration prior to running example.
 *
 * NOTE: When you start using Amazon SNS to send SMS messages, your AWS account
 * is in the SMS sandbox and you can only
 * use verified destination phone numbers. See https://docs.aws.amazon.com/sns/
 * latest/dg/sns-sms-sandbox.html.
 * NOTE: If destination is in the US, you also have an additional restriction
 * that you have use a dedicated
 * origination ID (phone number). You can request an origination number using
 * Amazon Pinpoint for a fee.
 * See https://aws.amazon.com/blogs/compute/provisioning-and-using-10dlc-
 * origination-numbers-with-amazon-sns/
 * for more information.
 *
 * <phone_number_value> input parameter uses E.164 format.
 * For example, in United States, this input value should be of the form:
 * +12223334444
 */
```

```
//! Send an SMS text message to a phone number.
/*!
 \param message: The message to publish.
 \param phoneNumber: The phone number of the recipient in E.164 format.
 \param clientConfiguration: AWS client configuration.
 \return bool: Function succeeded.
 */
bool AwsDoc::SNS::publishSms(const Aws::String &message,
                             const Aws::String &phoneNumber,
                             const Aws::Client::ClientConfiguration
&clientConfiguration) {
    Aws::SNS::SNSClient snsClient(clientConfiguration);

    Aws::SNS::Model::PublishRequest request;
    request.SetMessage(message);
    request.SetPhoneNumber(phoneNumber);

    const Aws::SNS::Model::PublishOutcome outcome = snsClient.Publish(request);

    if (outcome.IsSuccess()) {
        std::cout << "Message published successfully with message id, '"
                  << outcome.GetResult().GetMessageId() << "'."
                  << std::endl;
    }
    else {
        std::cerr << "Error while publishing message "
                  << outcome.GetError().GetMessage()
                  << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Para obter detalhes da API, consulte [Publish](#) na Referência da API do AWS SDK for C++ .

## Java

## SDK para Java 2.x

 Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.sns.SnsClient;
import software.amazon.awssdk.services.sns.model.PublishRequest;
import software.amazon.awssdk.services.sns.model.PublishResponse;
import software.amazon.awssdk.services.sns.model.SnsException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
 * started.html
 */
public class PublishTextSMS {
    public static void main(String[] args) {
        final String usage = ""

            Usage:    <message> <phoneNumber>

            Where:
                message - The message text to send.
                phoneNumber - The mobile phone number to which a message is
sent (for example, +1XXX5550100).\s
            """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }
    }
}
```

```
String message = args[0];
String phoneNumber = args[1];
SnsClient snsClient = SnsClient.builder()
    .region(Region.US_EAST_1)
    .build();
pubTextSMS(snsClient, message, phoneNumber);
snsClient.close();
}

public static void pubTextSMS(SnsClient snsClient, String message, String
phoneNumber) {
    try {
        PublishRequest request = PublishRequest.builder()
            .message(message)
            .phoneNumber(phoneNumber)
            .build();

        PublishResponse result = snsClient.publish(request);
        System.out
            .println(result.messageId() + " Message sent. Status was " +
result.sdkHttpResponse().statusCode());

    } catch (SnsException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}
}
```

- Para obter detalhes da API, consulte [Publish](#) na Referência da API do AWS SDK for Java 2.x .

## Kotlin

### SDK for Kotlin

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```
suspend fun pubTextSMS(messageVal: String?, phoneNumberVal: String?) {  
  
    val request = PublishRequest {  
        message = messageVal  
        phoneNumber = phoneNumberVal  
    }  
  
    SnsClient { region = "us-east-1" }.use { snsClient ->  
        val result = snsClient.publish(request)  
        println("${result.messageId} message sent.")  
    }  
}
```

- Para obter detalhes da API, consulte [Publish](#) na Referência da API do AWS SDK para Kotlin.

## PHP

### SDK para PHP

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
require 'vendor/autoload.php';  
  
use Aws\Exception\AwsException;  
use Aws\Sns\SnsClient;  
  
/**  
 * Sends a text message (SMS message) directly to a phone number using Amazon  
 * SNS.  
 *  
 * This code expects that you have AWS credentials set up per:  
 * https://docs.aws.amazon.com/sdk-for-php/v3/developer-guide/  
 * guide_credentials.html  
 */
```

```
$SnSClient = new SnsClient([
    'profile' => 'default',
    'region' => 'us-east-1',
    'version' => '2010-03-31'
]);

$message = 'This message is sent from a Amazon SNS code sample.';
$phone = '+1XXX5550100';

try {
    $result = $SnSClient->publish([
        'Message' => $message,
        'PhoneNumber' => $phone,
    ]);
    var_dump($result);
} catch (AwsException $e) {
    // output error message if fails
    error_log($e->getMessage());
}
```

- Para obter mais informações, consulte o [Guia do desenvolvedor do AWS SDK for PHP](#).
- Para obter detalhes da API, consulte [Publish](#) na Referência da API do AWS SDK for PHP .

## Python

### SDK para Python (Boto3)

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```
class SnsWrapper:
    """Encapsulates Amazon SNS topic and subscription functions."""

    def __init__(self, sns_resource):
        """
```

```
        :param sns_resource: A Boto3 Amazon SNS resource.
        """
        self.sns_resource = sns_resource

    def publish_text_message(self, phone_number, message):
        """
        Publishes a text message directly to a phone number without need for a
        subscription.

        :param phone_number: The phone number that receives the message. This
        must be
                               in E.164 format. For example, a United States phone
                               number might be +12065550101.
        :param message: The message to send.
        :return: The ID of the message.
        """
        try:
            response = self.sns_resource.meta.client.publish(
                PhoneNumber=phone_number, Message=message
            )
            message_id = response["MessageId"]
            logger.info("Published message to %s.", phone_number)
        except ClientError:
            logger.exception("Couldn't publish message to %s.", phone_number)
            raise
        else:
            return message_id
```

- Para obter detalhes da API, consulte [Publish](#) na Referência da API do AWS SDK for Python (Boto3).

## Monitorar e atividade de SMS

Ao monitorar a atividade do SMS, é possível manter o controle dos números de telefone de destino, das entregas bem-sucedidas ou com falha, dos motivos da falha, dos custos e de outras informações. O Amazon SNS ajuda resumindo estatísticas no console, enviando informações ao Amazon CloudWatch e enviando relatórios diários de uso de SMS para um bucket do Amazon S3 especificado por você.

## Tópicos

- [Visualizar estatísticas de entrega de SMS](#)
- [Visualizar métricas e logs do Amazon CloudWatch para entregas de SMS](#)
- [Visualizar relatórios de uso diário de SMS](#)

## Visualizar estatísticas de entrega de SMS

É possível usar o console do Amazon SNS para exibir estatísticas sobre suas entregas recentes de SMS.

1. Faça login no [console do Amazon SNS](#).
2. No menu do console, defina a seleção de região para uma [região que ofereça suporte a mensagens SMS](#).
3. No painel de navegação, escolha Text messaging (SMS) (Mensagens de texto (SMS)).
4. Na página Mensagens de texto (SMS), na seção Estatísticas da conta, veja os gráficos para suas entregas de mensagens SMS promocionais e transacionais. Cada gráfico mostra os seguintes dados para os 15 dias anteriores:
  - Taxa de entrega (porcentagem de entregas bem-sucedidas)
  - Enviadas (número de tentativas de entrega)
  - Falhas (número de falhas na entrega)

Nessa página, você também pode escolher o botão Usage (Uso) para acessar o bucket do Amazon S3 em que você armazena seus relatórios de uso diário. Para obter mais informações, consulte [Visualizar relatórios de uso diário de SMS](#).

## Visualizar métricas e logs do Amazon CloudWatch para entregas de SMS

É possível usar o Amazon CloudWatch e o Amazon CloudWatch Logs para monitorar as entregas de mensagens SMS.

## Tópicos

- [Visualizar métricas do Amazon CloudWatch](#)
- [Visualizar logs do CloudWatch](#)
- [Exemplo de log para entrega de SMS bem-sucedida](#)

- [Exemplo de log para entrega de SMS com falha](#)
- [Motivos para falha de entrega de SMS](#)

## Visualizar métricas do Amazon CloudWatch

O Amazon SNS coleta automaticamente as métricas sobre suas entregas de mensagens SMS e as envia ao Amazon CloudWatch. É possível usar o CloudWatch para monitorar essas métricas e criar alertas para avisá-lo quando uma métrica ultrapassar um limite. Por exemplo, você pode monitorar as métricas do CloudWatch para saber a taxa de entrega de SMS e seus encargos de SMS acumulados no mês.

Para obter informações sobre o monitoramento das métricas do CloudWatch, dos alarmes do CloudWatch e das métricas disponíveis, consulte [Monitorar tópicos do Amazon SNS usando o CloudWatch](#).

## Visualizar logs do CloudWatch

É possível coletar informações sobre entregas de mensagens SMS bem ou malsucedidas permitindo que o Amazon SNS grave dados no Amazon CloudWatch Logs. Para cada mensagem SMS que você enviar, o Amazon SNS gravará um log que inclui o preço da mensagem, o status de sucesso ou falha, o motivo da falha (se a mensagem falhou), o tempo de permanência da mensagem e outras informações.

Para habilitar e visualizar o CloudWatch Logs para suas mensagens SMS

1. Faça login no [console do Amazon SNS](#).
2. No menu do console, defina a seleção de região para uma [região que comporte mensagens SMS](#).
3. No painel de navegação, escolha Text messaging (SMS) [Mensagens de texto (SMS)].
4. Na página Mobile text messaging (SMS) [Mensagens de texto (SMS) em dispositivos móveis], na seção Text messaging preferences (Preferências de mensagens de texto), escolha Edit (Editar).
5. Na página seguinte, expanda a seção Delivery status logging (Registro do status da entrega).
6. Em Success sample rate (Taxa de amostragem de êxito), especifique a porcentagem de entregas de SMS bem-sucedidas para as quais o Amazon SNS gravará logs no CloudWatch Logs. Por exemplo:
  - Para gravar logs somente para entregas com falha, defina esse valor como 0.

- Para gravar logs para 10% de suas entregas bem-sucedidas, defina o valor como 10.

Se você não especificar uma porcentagem, o Amazon SNS gravará logs para todas as entregas bem-sucedidas.

7. Para fornecer as permissões necessárias, use uma das seguintes opções:

- Para criar uma nova função de serviço, escolha **Create new service role** (Criar nova função de serviço) e, em seguida, **Create new roles** (Criar novas funções). Na página seguinte, escolha **Allow** (Permitir) para que o Amazon SNS tenha acesso de gravação aos recursos de sua conta.
- Para usar uma função de serviço existente, escolha **Use existing service role** (Usar função de serviço existente) e, em seguida, cole o ARN na caixa **IAM role for successful and failed deliveries** (Função do IAM para entregas bem-sucedidas e com falha).

A função de serviço especificada deve permitir acesso de gravação aos recursos de sua conta. Para obter mais informações sobre como criar funções do IAM, consulte [Criar uma função para um serviço da AWS](#) no Guia do usuário do IAM.

8. Escolha **Save changes** (Salvar alterações).

9. Novamente na página **Mobile text messaging (SMS)** [Mensagens de texto para dispositivos móveis (SMS)], acesse a seção **Delivery status logs** (Logs de status de entrega) para exibir todos os logs disponíveis.

#### Note

Dependendo da operadora do número de telefone de destino, pode demorar até 72 horas para que os logs de entrega apareçam no console do Amazon SNS.

Exemplo de log para entrega de SMS bem-sucedida

O log de status de entrega para uma entrega de SMS bem-sucedida será semelhante ao exemplo a seguir:

```
{
  "notification": {
    "messageId": "34d9b400-c6dd-5444-820d-fbeb0f1f54cf",
    "timestamp": "2016-06-28 00:40:34.558"
  },
}
```

```
"delivery": {
  "phoneCarrier": "My Phone Carrier",
  "mnc": 270,
  "numberOfMessageParts": 1,
  "destination": "+1XXX5550100",
  "priceInUSD": 0.00645,
  "smsType": "Transactional",
  "mcc": 310,
  "providerResponse": "Message has been accepted by phone carrier",
  "dwellTimeMs": 599,
  "dwellTimeMsUntilDeviceAck": 1344
},
"status": "SUCCESS"
}
```

### Exemplo de log para entrega de SMS com falha

O log de status de entrega para uma entrega de SMS com falha será semelhante ao exemplo a seguir:

```
{
  "notification": {
    "messageId": "1077257a-92f3-5ca3-bc97-6a915b310625",
    "timestamp": "2016-06-28 00:40:34.559"
  },
  "delivery": {
    "mnc": 0,
    "numberOfMessageParts": 1,
    "destination": "+1XXX5550100",
    "priceInUSD": 0.00645,
    "smsType": "Transactional",
    "mcc": 0,
    "providerResponse": "Unknown error attempting to reach phone",
    "dwellTimeMs": 1420,
    "dwellTimeMsUntilDeviceAck": 1692
  },
  "status": "FAILURE"
}
```

### Motivos para falha de entrega de SMS

O motivo da falha é fornecido com o atributo `providerResponse`. As mensagens SMS podem não ser entregues pelos seguintes motivos:

- Bloqueada como spam pela operadora de telefonia
- O destino está em uma lista bloqueada
- Número de telefone inválido
- O corpo da mensagem é inválido
- A operadora de telefonia bloqueou essa mensagem
- A operadora de telefonia está inacessível/indisponível no momento
- O telefone bloqueou SMS
- O telefone está em uma lista bloqueada
- O telefone está inacessível/indisponível no momento
- O número de telefone solicitou o cancelamento do recebimento
- Essa entrega excede o preço máximo
- Erro desconhecido ao tentar entrar em contato com o telefone

## Visualizar relatórios de uso diário de SMS

É possível monitorar suas entregas de SMS inscrevendo-se para relatórios diários de uso do Amazon SNS. Para cada dia que você enviar pelo menos uma mensagem SMS, o Amazon SNS fornecerá um relatório de uso como um arquivo CSV para o bucket do Amazon S3 especificado. Leva 24 horas para que o relatório de uso de SMS esteja disponível no bucket do S3.

### Tópicos

- [Informações sobre relatório de uso diário](#)
- [Assinar relatórios de uso diário](#)

### Informações sobre relatório de uso diário

O relatório de uso inclui as seguintes informações para cada mensagem SMS que foi enviada de sua conta.

O relatório não inclui as mensagens que são enviadas aos destinatários que recusaram a opção.

- Hora da publicação da mensagem (em UTC)
- ID de mensagem
- Número de telefone de destino

- Tipo de mensagem
- Status da entrega
- Preço da mensagem (em USD)
- Número da parte (uma mensagem é dividida em várias partes se for muito longa para uma única mensagem)
- Número total de partes

 Note

Se o Amazon SNS não recebeu o número da peça, definimos seu valor como zero.

### Assinar relatórios de uso diário

Para inscrever-se para relatórios de uso diário, você deve criar um bucket do Amazon S3 com as permissões apropriadas.

Para criar um bucket do Amazon S3 para seus relatórios de uso diário

1. Da Conta da AWS que envia mensagens SMS, faça login no [console do Amazon S3](#).
2. Escolha Criar bucket.
3. Em Bucket Name (Nome do bucket), recomendamos inserir um nome exclusivo para sua conta e organização. Por exemplo, use o padrão <my-bucket-prefix>-<account\_id>-<org-id>.

Para obter informações sobre convenções e restrições de nomes de buckets, consulte [Regras de nomenclatura de buckets](#) no Guia do desenvolvedor do Amazon Simple Storage Service.

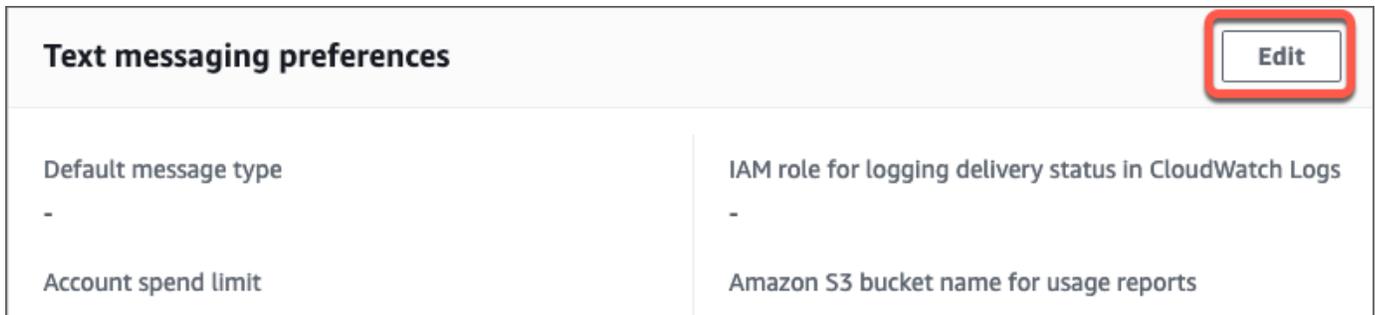
4. Escolha Create (Criar).
5. Na tabela All Buckets (Todos os buckets), escolha o bucket.
6. Na guia Permissions (Permissões), escolha Bucket policy (Política de bucket).
7. Na janela Bucket Policy Editor (Editor de política de bucket), forneça uma política que permita ao principal do serviço Amazon SNS gravar no bucket. Para ver um exemplo, consulte [Exemplo de política de bucket](#).

Se você usar a política de exemplo, lembre-se de substituir *my-s3-bucket* pelo nome do bucket escolhido na Etapa 3.

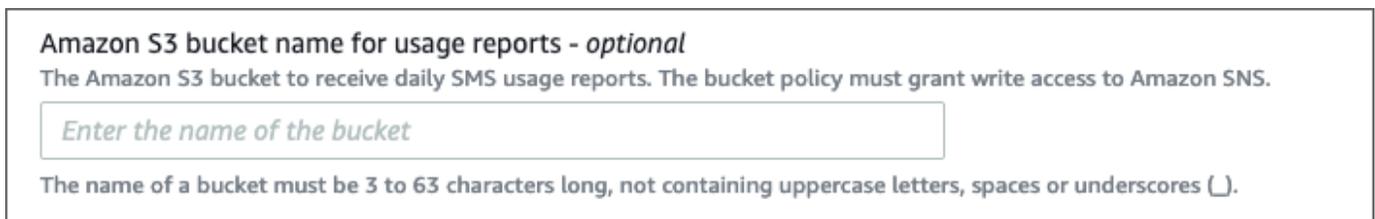
## 8. Escolha Save (Salvar).

Para inscrever-se para relatórios de uso diário

1. Faça login no [console do Amazon SNS](#).
2. No painel de navegação, escolha Text messaging (SMS) (Mensagens de texto (SMS)).
3. Na página Text messaging (SMS) (Mensagens de texto (SMS)), na seção Text messaging preferences (Preferências de mensagens de texto), escolha Edit (Editar).



4. Na página Edit text messaging preferences (Editar preferências de mensagens de texto), na seção Details (Detalhes), especifique o Amazon S3 bucket name for usage reports (Nome do bucket do Amazon S3 para relatórios de uso).



5. Selecione Save changes.

### Exemplo de política de bucket

A política a seguir permite que o principal do serviço Amazon SNS execute as ações `s3:PutObject`, `s3:GetBucketLocation` e `s3:ListBucket`.

A AWS fornece ferramentas para todos os serviços com entidades principais de serviço que receberam acesso aos recursos na sua conta. Quando a entidade principal em uma declaração de política do bucket do Amazon S3 é uma [AWS entidade principal do serviço da AWS](#), você pode usar as chaves de condição global `aws:SourceArn` ou `aws:SourceAccount` para proteção contra o [problema de representante confuso](#). Para limitar de qual região e conta o bucket pode receber relatórios de uso diário, use `aws:SourceArn`, como mostra o exemplo abaixo. Se não

quiser limitar quais regiões podem gerar esses relatórios, use `aws:SourceAccount` para limitar com base em qual conta está gerando os relatórios. Se você não conhece o ARN do recurso, use `aws:SourceAccount`.

Use o exemplo a seguir, que inclui proteção contra representante confuso quando você cria um bucket do Amazon S3 para receber relatórios de uso diário de SMS do Amazon SNS.

```
{
  "Version": "2008-10-17",
  "Statement": [{
    "Sid": "AllowPutObject",
    "Effect": "Allow",
    "Principal": {
      "Service": "sns.amazonaws.com"
    },
    "Action": "s3:PutObject",
    "Resource": "arn:aws:s3:::my-s3-bucket/*",
    "Condition": {
      "StringEquals": {
        "aws:SourceAccount": "account_id"
      },
      "ArnLike": {
        "aws:SourceArn": "arn:aws:sns:region:account_id:*"
      }
    }
  },
  {
    "Sid": "AllowGetBucketLocation",
    "Effect": "Allow",
    "Principal": {
      "Service": "sns.amazonaws.com"
    },
    "Action": "s3:GetBucketLocation",
    "Resource": "arn:aws:s3:::my-s3-bucket",
    "Condition": {
      "StringEquals": {
        "aws:SourceAccount": "account_id"
      },
      "ArnLike": {
        "aws:SourceArn": "arn:aws:sns:region:account_id:*"
      }
    }
  }
],
```

```

{
  "Sid": "AllowListBucket",
  "Effect": "Allow",
  "Principal": {
    "Service": "sns.amazonaws.com"
  },
  "Action": "s3:ListBucket",
  "Resource": "arn:aws:s3:::my-s3-bucket",
  "Condition": {
    "StringEquals": {
      "aws:SourceAccount": "account_id"
    },
    "ArnLike": {
      "aws:SourceArn": "arn:aws:sns:region:account_id:*"
    }
  }
}
]
}

```

### Note

É possível publicar relatórios de uso em buckets do Amazon S3 que pertencem à Conta da AWS especificada no elemento `Condition` na política do Amazon S3. Para publicar relatórios de uso em um bucket do Amazon S3 que pertence a outra Conta da AWS, consulte [Como posso copiar objetos S3 de outra Conta da AWS?](#)

## Exemplo de relatório de uso diário

Depois de inscrever-se nos relatórios de uso diários, a cada dia o Amazon SNS coloca um arquivo CSV com os dados de uso no seguinte local:

```
<my-s3-bucket>/SMSUsageReports/<region>/YYYY/MM/DD/00x.csv.gz
```

Cada arquivo pode conter até 50.000 registros. Se os registros de um dia excederem essa cota, o Amazon SNS adicionará vários arquivos.

A seguir, um exemplo de relatório:

```
PublishTimeUTC,MessageId,DestinationPhoneNumber,MessageType,DeliveryStatus,PriceInUSD,PartNumber
```

```
2016-05-10T03:00:29.476Z,96a298ac-1458-4825-
a7eb-7330e0720b72,1XXX5550100,Promotional,Message has been accepted by phone
carrier,0.90084,0,1
2016-05-10T03:00:29.561Z,1e29d394-
d7f4-4dc9-996e-26412032c344,1XXX5550100,Promotional,Message has been accepted by phone
carrier,0.34322,0,1
2016-05-10T03:00:30.769Z,98ba941c-afc7-4c51-
ba2c-56c6570a6c08,1XXX5550100,Transactional,Message has been accepted by phone
carrier,0.27815,0,1
```

## Gerenciar assinaturas de números de telefone e SMS

O Amazon SNS oferece várias opções para gerenciar quem recebe mensagens SMS de sua conta. Com uma frequência limitada, você poderá incluir números de telefone que cancelaram o recebimento de mensagens SMS de sua conta. Para interromper o envio de mensagens para inscrições de SMS, você pode remover as inscrições ou os tópicos que publicam neles.

### Tópicos

- [Cancelar recebimento de mensagens SMS](#)
- [Gerenciar números de telefone e assinaturas \(console\)](#)
- [Gerenciar números de telefone e assinaturas \(AWS SDKs\)](#)

## Cancelar recebimento de mensagens SMS

Quando exigido pelas leis e regulamentos locais (como nos Estados Unidos e Canadá), os destinatários de SMS podem usar seus dispositivos para cancelar a inscrição respondendo à mensagem com uma destas opções:

- ARRET (Francês)
- CANCEL
- END
- CANCELAR
- CANCELAR
- QUIT
- REMOVE
- PARAR
- TD

## • CANCELAR INSCRIÇÃO

Para cancelar o recebimento, o destinatário deve responder ao mesmo [número de origem](#) que o Amazon SNS usou para entregar a mensagem. Depois de cancelar, o destinatário não receberá mais mensagens SMS enviadas por você, a Conta da AWS menos que você opte pelo número de telefone.

Se o número de telefone estiver inscrito em um tópico do Amazon SNS, o cancelamento não removerá a inscrição, mas mensagens SMS apresentarão falha na entrega para essa inscrição, a menos que você inclua o número de telefone.

### Gerenciar números de telefone e assinaturas (console)

É possível usar o console do Amazon SNS para controlar quais números de telefone recebem mensagens SMS de sua conta.

#### Incluir um número de telefone que cancelou a assinatura

É possível ver quais números de telefone cancelaram o recebimento de mensagens SMS de sua conta, e incluí-los para retomar o envio de mensagens para eles.

Você pode incluir um número de telefone somente uma vez a cada 30 dias.

1. Faça login no [console do Amazon SNS](#).
2. No menu do console, defina a seleção de região para uma [região que comporte mensagens SMS](#).
3. No painel de navegação, escolha Text messaging (SMS) [Mensagens de texto (SMS)].
4. Na página Mensagens de texto (SMS), selecione Visualizar cancelamento de números de telefone. A página Números de telefone cancelados exibe os números de telefones cancelados.
5. Marque a caixa de seleção para o número de telefone que você deseja incluir e selecione Incluir. O número de telefone não terá mais o recebimento cancelado e receberá mensagens SMS que você enviar a ele.

#### Excluir uma assinatura de SMS

Exclua uma inscrição de SMS para interromper o envio de mensagens SMS para aquele número de telefone quando você publicar nos tópicos.

1. No painel de navegação, escolha Subscriptions (Assinaturas).

2. Marque as caixas de seleção para as inscrições que você deseja excluir. Escolha Ações e escolha Excluir inscrições.
3. Na janela Delete (Excluir), escolha Delete (Excluir). O Amazon SNS exclui a assinatura e exibe uma mensagem de êxito.

## Excluir um tópico

Exclua um tópico quando você não desejar mais publicar mensagens em seus endpoints inscritos.

1. No painel de navegação, escolha Topics (Tópicos).
2. Marque as caixas de seleção para os tópicos que você deseja excluir. Escolha Ações e escolha Excluir tópicos.
3. Na janela Delete (Excluir), escolha Delete (Excluir). O Amazon SNS exclui o tópico e exibe uma mensagem de êxito.

## Gerenciar números de telefone e assinaturas (AWS SDKs)

Você pode usar os AWS SDKs para fazer solicitações programáticas ao Amazon SNS e gerenciar quais números de telefone podem receber mensagens SMS da sua conta.

Para usar um AWS SDK, você deve configurá-lo com suas credenciais. Para obter mais informações, consulte [Shared config and credentials files](#) (“Arquivos compartilhados de configuração e de credenciais”) no Guia de referência de AWS SDKs e ferramentas.

### Visualizar todos os números de telefone que cancelaram o recebimento

Para visualizar todos os números de telefone cancelados, envie uma solicitação `ListPhoneNumbersOptedOut` com a API do Amazon SNS.

Os exemplos de código a seguir mostram como usar `ListPhoneNumbersOptedOut`.

## CLI

### AWS CLI

Para listar as opções de cancelamento de mensagens SMS

O exemplo `list-phone-numbers-opted-out` a seguir lista os números de telefone que optaram por não receber mensagens SMS.

```
aws sns list-phone-numbers-opted-out
```

Saída:

```
{
  "phoneNumbers": [
    "+15555550100"
  ]
}
```

- Para obter detalhes da API, consulte [ListPhoneNumbersOptedOut](#) em Referência de AWS CLI Comandos.

Java

SDK para Java 2.x

 Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.sns.SnsClient;
import software.amazon.awssdk.services.sns.model.ListPhoneNumbersOptedOutRequest;
import
  software.amazon.awssdk.services.sns.model.ListPhoneNumbersOptedOutResponse;
import software.amazon.awssdk.services.sns.model.SnsException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
 * started.html
 */
public class ListOptOut {
```

```
public static void main(String[] args) {
    SnsClient snsClient = SnsClient.builder()
        .region(Region.US_EAST_1)
        .build();

    listOpts(snsClient);
    snsClient.close();
}

public static void listOpts(SnsClient snsClient) {
    try {
        ListPhoneNumbersOptedOutRequest request =
ListPhoneNumbersOptedOutRequest.builder().build();
        ListPhoneNumbersOptedOutResponse result =
snsClient.listPhoneNumbersOptedOut(request);
        System.out.println("Status is " +
result.sdkHttpResponse().statusCode() + "\n\nPhone Numbers: \n\n"
            + result.phoneNumbers());

    } catch (SnsException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}
}
```

- Para obter detalhes da API, consulte [ListPhoneNumbersOptedOut](#) Referência AWS SDK for Java 2.x da API.

## PHP

### SDK para PHP

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
require 'vendor/autoload.php';
```

```
use Aws\Exception\AwsException;
use Aws\Sns\SnsClient;

/**
 * Returns a list of phone numbers that are opted out of receiving SMS messages
 * from your AWS SNS account.
 *
 * This code expects that you have AWS credentials set up per:
 * https://docs.aws.amazon.com/sdk-for-php/v3/developer-guide/
 * guide_credentials.html
 */

$SnSClient = new SnsClient([
    'profile' => 'default',
    'region' => 'us-east-1',
    'version' => '2010-03-31'
]);

try {
    $result = $SnSClient->listPhoneNumbersOptedOut();
    var_dump($result);
} catch (AwsException $e) {
    // output error message if fails
    error_log($e->getMessage());
}
```

- Para obter mais informações, consulte o [Guia do desenvolvedor do AWS SDK for PHP](#).
- Para obter detalhes da API, consulte [ListPhoneNumbersOptedOut](#) Referência AWS SDK for PHP da API.

## Verificar o cancelamento de um número de telefone

Para verificar se um número de telefone cancelou o recebimento, envie uma solicitação `CheckIfPhoneNumberIsOptedOut` com a API do Amazon SNS.

Os exemplos de código a seguir mostram como usar `CheckIfPhoneNumberIsOptedOut`.

## .NET

### AWS SDK for .NET

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```
using System;
using System.Threading.Tasks;
using Amazon.SimpleNotificationService;
using Amazon.SimpleNotificationService.Model;

/// <summary>
/// This example shows how to use the Amazon Simple Notification Service
/// (Amazon SNS) to check whether a phone number has been opted out.
/// </summary>
public class IsPhoneNumOptedOut
{
    public static async Task Main()
    {
        string phoneNumber = "+15551112222";

        IAmazonSimpleNotificationService client = new
AmazonSimpleNotificationServiceClient();

        await CheckIfOptedOutAsync(client, phoneNumber);
    }

    /// <summary>
    /// Checks to see if the supplied phone number has been opted out.
    /// </summary>
    /// <param name="client">The initialized Amazon SNS Client object used
    /// to check if the phone number has been opted out.</param>
    /// <param name="phoneNumber">A string representing the phone number
    /// to check.</param>
    public static async Task
CheckIfOptedOutAsync(IAmazonSimpleNotificationService client, string
phoneNumber)
    {
```

```
var request = new CheckIfPhoneNumberIsOptedOutRequest
{
    PhoneNumber = phoneNumber,
};

try
{
    var response = await
client.CheckIfPhoneNumberIsOptedOutAsync(request);

    if (response.HttpStatusCode == System.Net.HttpStatusCode.OK)
    {
        string optOutStatus = response.IsOptedOut ? "opted out" :
"not opted out.";
        Console.WriteLine($"The phone number: {phoneNumber} is
{optOutStatus}");
    }
    catch (AuthorizationErrorException ex)
    {
        Console.WriteLine($"{ex.Message}");
    }
}
}
```

- Para obter detalhes da API, consulte [CheckIfPhoneNumberIsOptedOut](#) Referência AWS SDK for .NET da API.

## CLI

### AWS CLI

Para verificar o cancelamento de mensagens SMS para um número de telefone

O `check-if-phone-number-is-opted-out` exemplo a seguir verifica se o número de telefone especificado optou por não receber mensagens SMS da AWS conta atual.

```
aws sns check-if-phone-number-is-opted-out \
  --phone-number +1555550100
```

**Saída:**

```
{
  "isOptedOut": false
}
```

- Para obter detalhes da API, consulte [CheckIfPhoneNumberIsOptedOut](#) em Referência de AWS CLI Comandos.

**Java****SDK para Java 2.x****Note**

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.sns.SnsClient;
import
  software.amazon.awssdk.services.sns.model.CheckIfPhoneNumberIsOptedOutRequest;
import
  software.amazon.awssdk.services.sns.model.CheckIfPhoneNumberIsOptedOutResponse;
import software.amazon.awssdk.services.sns.model.SnsException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
 * started.html
 */
public class CheckOptOut {
    public static void main(String[] args) {

        final String usage = ""
```

```
Usage:    <phoneNumber>

Where:
    phoneNumber - The mobile phone number to look up (for example,
+1XXX5550100).

""";

if (args.length != 1) {
    System.out.println(usage);
    System.exit(1);
}

String phoneNumber = args[0];
SnsClient snsClient = SnsClient.builder()
    .region(Region.US_EAST_1)
    .build();

checkPhone(snsClient, phoneNumber);
snsClient.close();
}

public static void checkPhone(SnsClient snsClient, String phoneNumber) {
    try {
        CheckIfPhoneNumberIsOptedOutRequest request =
CheckIfPhoneNumberIsOptedOutRequest.builder()
            .phoneNumber(phoneNumber)
            .build();

        CheckIfPhoneNumberIsOptedOutResponse result =
snsClient.checkIfPhoneNumberIsOptedOut(request);
        System.out.println(
            result.isOptedOut() + "Phone Number " + phoneNumber + " has
Opted Out of receiving sns messages." +
                "\n\nStatus was " +
result.sdkHttpResponse().statusCode());

    } catch (SnsException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}
}
```

- Para obter detalhes da API, consulte [CheckIfPhoneNumberIsOptedOut](#) Referência AWS SDK for Java 2.x da API.

## JavaScript

### SDK para JavaScript (v3)

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

Crie o cliente em um módulo separado e exporte-o.

```
import { SNSClient } from "@aws-sdk/client-sns";

// The AWS Region can be provided here using the `region` property. If you leave
// it blank
// the SDK will default to the region set in your AWS config.
export const snsClient = new SNSClient({});
```

Importe o SDK e os módulos do cliente e chame a API.

```
import { CheckIfPhoneNumberIsOptedOutCommand } from "@aws-sdk/client-sns";

import { snsClient } from "../libs/snsClient.js";

export const checkIfPhoneNumberIsOptedOut = async (
  phoneNumber = "5555555555",
) => {
  const command = new CheckIfPhoneNumberIsOptedOutCommand({
    phoneNumber,
  });

  const response = await snsClient.send(command);
  console.log(response);
```

```
// {
//   '$metadata': {
//     httpStatusCode: 200,
//     requestId: '3341c28a-cdc8-5b39-a3ee-9fb0ee125732',
//     extendedRequestId: undefined,
//     cfId: undefined,
//     attempts: 1,
//     totalRetryDelay: 0
//   },
//   isOptedOut: false
// }
return response;
};
```

- Para obter mais informações, consulte o [Guia do desenvolvedor do AWS SDK for JavaScript](#).
- Para obter detalhes da API, consulte [CheckIfPhoneNumberIsOptedOut](#) Referência AWS SDK for JavaScript da API.

## PHP

### SDK para PHP

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
require 'vendor/autoload.php';

use Aws\Exception\AwsException;
use Aws\Sns\SnsClient;

/**
 * Indicates whether the phone number owner has opted out of receiving SMS
 * messages from your AWS SNS account.
 *
 * This code expects that you have AWS credentials set up per:
```

```
* https://docs.aws.amazon.com/sdk-for-php/v3/developer-guide/  
guide_credentials.html  
*/  
  
$SnSClient = new SnsClient([  
    'profile' => 'default',  
    'region' => 'us-east-1',  
    'version' => '2010-03-31'  
]);  
  
$phone = '+1XXX5550100';  
  
try {  
    $result = $SnSClient->checkIfPhoneNumberIsOptedOut([  
        'phoneNumber' => $phone,  
    ]);  
    var_dump($result);  
} catch (AwsException $e) {  
    // output error message if fails  
    error_log($e->getMessage());  
}
```

- Para obter mais informações, consulte o [Guia do desenvolvedor do AWS SDK for PHP](#).
- Para obter detalhes da API, consulte [CheckIfPhoneNumberIsOptedOuta](#) Referência AWS SDK for PHP da API.

Incluir um número de telefone que cancelou a assinatura

Para incluir um número de telefone que cancelou o recebimento, envie uma solicitação `OptInPhoneNumber` com a API do Amazon SNS.

Você pode incluir um número de telefone somente uma vez a cada 30 dias.

Excluir uma assinatura de SMS

Para excluir uma inscrição de SMS de um tópico do Amazon SNS, obtenha o ARN da inscrição enviando uma solicitação `ListSubscriptions` com a API do Amazon SNS e, em seguida, informe o ARN em uma solicitação `Unsubscribe`.

Os exemplos de código a seguir mostram como usar `Unsubscribe`.

## .NET

### AWS SDK for .NET

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

Cancele a assinatura de um tópico por meio de um ARN de assinatura.

```
/// <summary>
/// Unsubscribe from a topic by a subscription ARN.
/// </summary>
/// <param name="subscriptionArn">The ARN of the subscription.</param>
/// <returns>True if successful.</returns>
public async Task<bool> UnsubscribeByArn(string subscriptionArn)
{
    var unsubscribeResponse = await _amazonSNSClient.UnsubscribeAsync(
        new UnsubscribeRequest()
        {
            SubscriptionArn = subscriptionArn
        });
    return unsubscribeResponse.HttpStatusCode == HttpStatusCode.OK;
}
```

- Para obter detalhes da API, consulte [Unsubscribe](#) na Referência da API AWS SDK for .NET

## C++

### SDK para C++

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```

//! Delete a subscription to an Amazon Simple Notification Service (Amazon SNS)
  topic.
/*!
  \param subscriptionARN: The Amazon Resource Name (ARN) for an Amazon SNS topic
  subscription.
  \param clientConfiguration: AWS client configuration.
  \return bool: Function succeeded.
*/
bool AwsDoc::SNS::unsubscribe(const Aws::String &subscriptionARN,
                              const Aws::Client::ClientConfiguration
&clientConfiguration) {
    Aws::SNS::SNSClient snsClient(clientConfiguration);

    Aws::SNS::Model::UnsubscribeRequest request;
    request.SetSubscriptionArn(subscriptionARN);

    const Aws::SNS::Model::UnsubscribeOutcome outcome =
snsClient.Unsubscribe(request);

    if (outcome.IsSuccess()) {
        std::cout << "Unsubscribed successfully " << std::endl;
    }
    else {
        std::cerr << "Error while unsubscribing " <<
outcome.GetError().GetMessage()
        << std::endl;
    }

    return outcome.IsSuccess();
}

```

- Para obter detalhes da API, consulte [Unsubscribe](#) na Referência da API AWS SDK for C++ .

## CLI

### AWS CLI

Para cancelar a assinatura de um tópico

O exemplo `unsubscribe` a seguir exclui a assinatura especificada de um tópico.

```
aws sns unsubscribe \  
  --subscription-arn arn:aws:sns:us-west-2:0123456789012:my-  
  topic:8a21d249-4329-4871-acc6-7be709c6ea7f
```

Este comando não produz saída.

- Para obter detalhes da API, consulte [Unsubscribe](#) na Referência de comandos da AWS CLI.

## Java

### SDK para Java 2.x

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```
import software.amazon.awssdk.regions.Region;  
import software.amazon.awssdk.services.sns.SnsClient;  
import software.amazon.awssdk.services.sns.model.SnsException;  
import software.amazon.awssdk.services.sns.model.UnsubscribeRequest;  
import software.amazon.awssdk.services.sns.model.UnsubscribeResponse;  
  
/**  
 * Before running this Java V2 code example, set up your development  
 * environment, including your credentials.  
 *  
 * For more information, see the following documentation topic:  
 *  
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html  
 */  
public class Unsubscribe {  
    public static void main(String[] args) {  
        final String usage = ""  
  
            Usage:    <subscriptionArn>  
  
            Where:
```

```
        subscriptionArn - The ARN of the subscription to delete.
        """);

    if (args.length < 1) {
        System.out.println(usage);
        System.exit(1);
    }

    String subscriptionArn = args[0];
    SnsClient snsClient = SnsClient.builder()
        .region(Region.US_EAST_1)
        .build();

    unSub(snsClient, subscriptionArn);
    snsClient.close();
}

public static void unSub(SnsClient snsClient, String subscriptionArn) {
    try {
        UnsubscribeRequest request = UnsubscribeRequest.builder()
            .subscriptionArn(subscriptionArn)
            .build();

        UnsubscribeResponse result = snsClient.unsubscribe(request);
        System.out.println("\n\nStatus was " +
            result.sdkHttpResponse().statusCode()
            + "\n\nSubscription was removed for " +
            request.subscriptionArn());

    } catch (SnsException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}
}
```

- Para obter detalhes da API, consulte [Unsubscribe](#) na Referência da API AWS SDK for Java 2.x .

## JavaScript

### SDK para JavaScript (v3)

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

Crie o cliente em um módulo separado e exporte-o.

```
import { SNSClient } from "@aws-sdk/client-sns";

// The AWS Region can be provided here using the `region` property. If you leave
// it blank
// the SDK will default to the region set in your AWS config.
export const snsClient = new SNSClient({});
```

Importe o SDK e os módulos do cliente e chame a API.

```
import { UnsubscribeCommand } from "@aws-sdk/client-sns";
import { snsClient } from "../libs/snsClient.js";

/**
 * @param {string} subscriptionArn - The ARN of the subscription to cancel.
 */
const unsubscribe = async (
  subscriptionArn = "arn:aws:sns:us-east-1:xxxxxxxxxxxx:mytopic:xxxxxxxx-xxxx-
  xxxx-xxxx-xxxxxxxxxxxx",
) => {
  const response = await snsClient.send(
    new UnsubscribeCommand({
      SubscriptionArn: subscriptionArn,
    }),
  );
  console.log(response);
  // {
  //   '$metadata': {
  //     httpStatusCode: 200,
  //     requestId: '0178259a-9204-507c-b620-78a7570a44c6',
```

```
//     extendedRequestId: undefined,  
//     cfId: undefined,  
//     attempts: 1,  
//     totalRetryDelay: 0  
//   }  
// }  
return response;  
};
```

- Para obter mais informações, consulte o [Guia do desenvolvedor do AWS SDK for JavaScript](#).
- Para obter detalhes da API, consulte [Unsubscribe](#) na Referência da API AWS SDK for JavaScript .

## Kotlin

### SDK for Kotlin

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```
suspend fun unSub(subscriptionArnVal: String) {  
  
    val request = UnsubscribeRequest {  
        subscriptionArn = subscriptionArnVal  
    }  
  
    SnsClient { region = "us-east-1" }.use { snsClient ->  
        snsClient.unsubscribe(request)  
        println("Subscription was removed for ${request.subscriptionArn}")  
    }  
}
```

- Para obter detalhes da API, consulte [Cancelar assinatura](#) na Referência da API AWS SDK para Kotlin.

## PHP

## SDK para PHP

 Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
require 'vendor/autoload.php';

use Aws\Exception\AwsException;
use Aws\Sns\SnsClient;

/**
 * Deletes a subscription to an Amazon SNS topic.
 *
 * This code expects that you have AWS credentials set up per:
 * https://docs.aws.amazon.com/sdk-for-php/v3/developer-guide/
 * guide_credentials.html
 */

$SnsClient = new SnsClient([
    'profile' => 'default',
    'region' => 'us-east-1',
    'version' => '2010-03-31'
]);

$subscription = 'arn:aws:sns:us-east-1:111122223333:MySubscription';

try {
    $result = $SnsClient->unsubscribe([
        'SubscriptionArn' => $subscription,
    ]);
    var_dump($result);
} catch (AwsException $e) {
    // output error message if fails
    error_log($e->getMessage());
}
```

- Para obter mais informações, consulte o [Guia do desenvolvedor do AWS SDK for PHP](#).
- Para obter detalhes da API, consulte [Unsubscribe](#) na Referência da API AWS SDK for PHP

## Python

### SDK para Python (Boto3).

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```
class SnsWrapper:
    """Encapsulates Amazon SNS topic and subscription functions."""

    def __init__(self, sns_resource):
        """
        :param sns_resource: A Boto3 Amazon SNS resource.
        """
        self.sns_resource = sns_resource

    @staticmethod
    def delete_subscription(subscription):
        """
        Unsubscribes and deletes a subscription.
        """
        try:
            subscription.delete()
            logger.info("Deleted subscription %s.", subscription.arn)
        except ClientError:
            logger.exception("Couldn't delete subscription %s.",
                subscription.arn)
            raise
```

- Para obter detalhes da API, consulte [Unsubscribe](#) na Referência da API AWS SDK for Python (Boto3).

## SAP ABAP

### SDK para SAP ABAP

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```
TRY.  
    lo_sns->unsubscribe( iv_subscriptionarn = iv_subscription_arn ).  
    MESSAGE 'Subscription deleted.' TYPE 'I'.  
CATCH /aws1/cx_snsnotfoundexception.  
    MESSAGE 'Subscription does not exist.' TYPE 'E'.  
CATCH /aws1/cx_snsinvalidparameterex.  
    MESSAGE 'Subscription with "PendingConfirmation" status cannot be  
deleted/unsubscribed. Confirm subscription before performing unsubscribe  
operation.' TYPE 'E'.  
ENDTRY.
```

- Para obter detalhes da API, consulte [Cancelar assinatura](#) na Referência da API do AWS SDK para SAP ABAP.

## Excluir um tópico

Para excluir um tópico e todas as suas inscrições, obtenha o ARN do tópico enviando uma solicitação `ListTopics` com a API do Amazon SNS e, em seguida, informe o ARN em uma solicitação `DeleteTopic`.

Os exemplos de código a seguir mostram como usar `DeleteTopic`.

## .NET

### AWS SDK for .NET

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

Exclua um tópico por meio do respectivo ARN.

```
/// <summary>
/// Delete a topic by its topic ARN.
/// </summary>
/// <param name="topicArn">The ARN of the topic.</param>
/// <returns>True if successful.</returns>
public async Task<bool> DeleteTopicByArn(string topicArn)
{
    var deleteResponse = await _amazonSNSClient.DeleteTopicAsync(
        new DeleteTopicRequest()
        {
            TopicArn = topicArn
        });
    return deleteResponse.HttpStatusCode == HttpStatusCode.OK;
}
```

- Para obter detalhes da API, consulte [DeleteTópico](#) Referência AWS SDK for .NET da API.

## C++

### SDK para C++

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```
#!/ Delete an Amazon Simple Notification Service (Amazon SNS) topic.
/*!
  \param topicARN: The Amazon Resource Name (ARN) for an Amazon SNS topic.
  \param clientConfiguration: AWS client configuration.
  \return bool: Function succeeded.
*/
bool AwsDoc::SNS::deleteTopic(const Aws::String &topicARN,
                              const Aws::Client::ClientConfiguration
&clientConfiguration) {
    Aws::SNS::SNSClient snsClient(clientConfiguration);

    Aws::SNS::Model::DeleteTopicRequest request;
    request.SetTopicArn(topicARN);

    const Aws::SNS::Model::DeleteTopicOutcome outcome =
snsClient.DeleteTopic(request);

    if (outcome.IsSuccess()) {
        std::cout << "Successfully deleted the Amazon SNS topic " << topicARN <<
std::endl;
    }
    else {
        std::cerr << "Error deleting topic " << topicARN << ":" <<
outcome.GetError().GetMessage() << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Para obter detalhes da API, consulte [DeleteTopic](#) Referência AWS SDK for C++ da API.

## CLI

### AWS CLI

Para excluir um tópico do SNS

O exemplo `delete-topic` a seguir exclui o tópico do SNS especificado.

```
aws sns delete-topic \
  --topic-arn "arn:aws:sns:us-west-2:123456789012:my-topic"
```

Este comando não produz saída.

- Para obter detalhes da API, consulte [DeleteTopic](#) em Referência de AWS CLI Comandos.

Go

SDK for Go V2

 Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```
// SnsActions encapsulates the Amazon Simple Notification Service (Amazon SNS)
// actions
// used in the examples.
type SnsActions struct {
    SnsClient *sns.Client
}

// DeleteTopic delete an Amazon SNS topic.
func (actor SnsActions) DeleteTopic(topicArn string) error {
    _, err := actor.SnsClient.DeleteTopic(context.TODO(), &sns.DeleteTopicInput{
        TopicArn: aws.String(topicArn)})
    if err != nil {
        log.Printf("Couldn't delete topic %v. Here's why: %v\n", topicArn, err)
    }
    return err
}
```

- Para obter detalhes da API, consulte [DeleteTopic](#) em Referência AWS SDK for Go da API.

## Java

## SDK para Java 2.x

 Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.sns.SnsClient;
import software.amazon.awssdk.services.sns.model.DeleteTopicRequest;
import software.amazon.awssdk.services.sns.model.DeleteTopicResponse;
import software.amazon.awssdk.services.sns.model.SnsException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
 * started.html
 */
public class DeleteTopic {
    public static void main(String[] args) {
        final String usage = ""

                Usage:      <topicArn>

                Where:
                    topicArn - The ARN of the topic to delete.
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String topicArn = args[0];
        SnsClient snsClient = SnsClient.builder()
```

```
        .region(Region.US_EAST_1)
        .build();

        System.out.println("Deleting a topic with name: " + topicArn);
        deleteSNSTopic(snsClient, topicArn);
        snsClient.close();
    }

    public static void deleteSNSTopic(SnsClient snsClient, String topicArn) {
        try {
            DeleteTopicRequest request = DeleteTopicRequest.builder()
                .topicArn(topicArn)
                .build();

            DeleteTopicResponse result = snsClient.deleteTopic(request);
            System.out.println("\n\nStatus was " +
                result.sdkHttpResponse().statusCode());

        } catch (SnsException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Para obter detalhes da API, consulte [DeleteTopica](#) Referência AWS SDK for Java 2.x da API.

## JavaScript

### SDK para JavaScript (v3)

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

Crie o cliente em um módulo separado e exporte-o.

```
import { SNSClient } from "@aws-sdk/client-sns";
```

```
// The AWS Region can be provided here using the `region` property. If you leave
it blank
// the SDK will default to the region set in your AWS config.
export const snsClient = new SNSClient({});
```

Importe o SDK e os módulos do cliente e chame a API.

```
import { DeleteTopicCommand } from "@aws-sdk/client-sns";
import { snsClient } from "../libs/snsClient.js";

/**
 * @param {string} topicArn - The ARN of the topic to delete.
 */
export const deleteTopic = async (topicArn = "TOPIC_ARN") => {
  const response = await snsClient.send(
    new DeleteTopicCommand({ TopicArn: topicArn }),
  );
  console.log(response);
  // {
  //   '$metadata': {
  //     httpStatusCode: 200,
  //     requestId: 'a10e2886-5a8f-5114-af36-75bd39498332',
  //     extendedRequestId: undefined,
  //     cfId: undefined,
  //     attempts: 1,
  //     totalRetryDelay: 0
  //   }
  // }
};
```

- Para obter mais informações, consulte o [Guia do desenvolvedor do AWS SDK for JavaScript](#).
- Para obter detalhes da API, consulte [DeleteTopic](#) Referência AWS SDK for JavaScript da API.

## Kotlin

### SDK for Kotlin

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```
suspend fun deleteSNSTopic(topicArnVal: String) {  
  
    val request = DeleteTopicRequest {  
        topicArn = topicArnVal  
    }  
  
    SnsClient { region = "us-east-1" }.use { snsClient ->  
        snsClient.deleteTopic(request)  
        println("$topicArnVal was successfully deleted.")  
    }  
}
```

- Para obter detalhes da API, consulte a [DeleteTopic](#) referência da API AWS SDK for Kotlin.

## PHP

### SDK para PHP

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```
require 'vendor/autoload.php';  
  
use Aws\Exception\AwsException;  
use Aws\Sns\SnsClient;
```

```
/**
 * Deletes an SNS topic and all its subscriptions.
 *
 * This code expects that you have AWS credentials set up per:
 * https://docs.aws.amazon.com/sdk-for-php/v3/developer-guide/
guide_credentials.html
 */

$SnSClient = new SnsClient([
    'profile' => 'default',
    'region' => 'us-east-1',
    'version' => '2010-03-31'
]);

$topic = 'arn:aws:sns:us-east-1:111122223333:MyTopic';

try {
    $result = $SnSClient->deleteTopic([
        'TopicArn' => $topic,
    ]);
    var_dump($result);
} catch (AwsException $e) {
    // output error message if fails
    error_log($e->getMessage());
}
```

- Para obter detalhes da API, consulte [DeleteTopic](#) Referência AWS SDK for PHP da API.

## Python

SDK para Python (Boto3).

### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```
class SnsWrapper:
```

```
"""Encapsulates Amazon SNS topic and subscription functions."""

def __init__(self, sns_resource):
    """
    :param sns_resource: A Boto3 Amazon SNS resource.
    """
    self.sns_resource = sns_resource

    @staticmethod
    def delete_topic(topic):
        """
        Deletes a topic. All subscriptions to the topic are also deleted.
        """
        try:
            topic.delete()
            logger.info("Deleted topic %s.", topic.arn)
        except ClientError:
            logger.exception("Couldn't delete topic %s.", topic.arn)
            raise
```

- Para obter detalhes da API, consulte a [DeleteTopic](#) Referência da API AWS SDK for Python (Boto3).

## SAP ABAP

### SDK para SAP ABAP

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```
TRY.
  lo_sns->deletetopic( iv_topicarn = iv_topic_arn ).
  MESSAGE 'SNS topic deleted.' TYPE 'I'.
CATCH /aws1/cx_snsnotfoundexception.
  MESSAGE 'Topic does not exist.' TYPE 'E'.
```

```
ENDTRY.
```

- Para obter detalhes da API, consulte a [DeleteTopic](#) referência da API AWS SDK for SAP ABAP.

## Países e regiões compatíveis

### Important

Em 31 de agosto de 2023, um número dedicado, como um número [10DLC](#), ou um [número gratuito](#) passou a ser necessário para enviar mensagens de texto SMS para os Estados Unidos e os respectivos territórios (Guam, Porto Rico, Ilhas Samoa Americana e Ilhas Virgens dos US).

Atualmente, o Amazon SNS oferece suporte a mensagens SMS nas seguintes regiões: AWS

Nome da região	Região	Endpoint	Protocolo
Leste dos EUA (Ohio)	us-east-2	sns.us-east-2.amazonaws.com	HTTP e HTTPS
Leste dos EUA (Norte da Virgínia)	us-east-1	sns.us-east-1.amazonaws.com	HTTP e HTTPS
Oeste dos EUA (Norte da Califórnia)	us-west-1	sns.us-west-1.amazonaws.com	HTTP e HTTPS
Oeste dos EUA (Oregon)	us-west-2	sns.us-west-2.amazonaws.com	HTTP e HTTPS
África (Cidade do Cabo)	af-south-1	sns.af-south-1.amazonaws.com	HTTP e HTTPS
Ásia-Pacífico (Hyderabad)	ap-south-2	sns.ap-south-2.amazonaws.com	HTTP e HTTPS

Nome da região	Região	Endpoint	Protocolo
Ásia-Pacífico (Jacarta)	ap-southeast-3	sns.ap-southeast-3 .amazonaws.com	HTTP e HTTPS
Ásia-Pacífico (Melbourne)	ap-southeast-4	sns.ap-southeast-4 .amazonaws.com	HTTP e HTTPS
Ásia-Pacífico (Mumbai)	ap-south-1	sns.ap-south-1.ama zonaws.com	HTTP e HTTPS
Asia Pacific (Osaka)	ap-northeast-3	sns.ap-northeast-3 .amazonaws.com	HTTP e HTTPS
Ásia-Pacífico (Singapura)	ap-southeast-1	sns.ap-southeast-1 .amazonaws.com	HTTP e HTTPS
Ásia-Pacífico (Sydney)	ap-southeast-2	sns.ap-southeast-2 .amazonaws.com	HTTP e HTTPS
Ásia-Pacífico (Tóquio)	ap-northeast-1	sns.ap-northeast-1 .amazonaws.com	HTTP e HTTPS
Canadá (Central)	ca-central-1	sns.ca-central-1.a mazonaws.com	HTTP e HTTPS
Europa (Frankfurt)	eu-central-1	sns.eu-central-1.a mazonaws.com	HTTP e HTTPS
Europa (Irlanda)	eu-west-1	sns.eu-west-1.amaz onaws.com	HTTP e HTTPS
Europa (Londres)	eu-west-2	sns.eu-west-2.amaz onaws.com	HTTP e HTTPS
Europa (Milão)	eu-south-1	sns.eu-south-1.ama zonaws.com	HTTP e HTTPS
Europa (Paris)	eu-west-3	sns.eu-west-3.amaz onaws.com	HTTP e HTTPS

Nome da região	Região	Endpoint	Protocolo
Europa (Espanha)	eu-south-2	sns.eu-south-2.amazonaws.com	HTTP e HTTPS
Europa (Estocolmo)	eu-north-1	sns.eu-north-1.amazonaws.com	HTTP e HTTPS
Europa (Zurique)	eu-central-2	sns.eu-central-2.amazonaws.com	HTTP e HTTPS
Israel (Tel Aviv)	il-central-1	sns.il-central-1.amazonaws.com	HTTP e HTTPS
Oriente Médio (Barém)	me-south-1	sns.me-south-1.amazonaws.com	HTTP e HTTPS
Oriente Médio (Emirados Árabes Unidos)	me-central-1	sns.me-central-1.amazonaws.com	HTTP e HTTPS
América do Sul (São Paulo)	sa-east-1	sns.sa-east-1.amazonaws.com	HTTP e HTTPS
AWS GovCloud (Leste dos EUA)	us-gov-east-1	sns.us-gov-east-1.amazonaws.com	HTTP e HTTPS
AWS GovCloud (Oeste dos EUA)	us-gov-west-1	sns.us-gov-west-1.amazonaws.com	HTTP e HTTPS

O Amazon SNS pode ser usado para enviar mensagens SMS para os seguintes países e regiões:

 Note

O uso de IDs de remetente em países com suporte pode melhorar a entrega de SMS.

País ou região	Código ISO	Código de discagem	É compatível com códigos curtos	É compatível com códigos longos	Oferece suporte a IDs de remetente	Compatível com SMS bidirecional
<b>A</b>						
Afeganistão	AF	93	Não	Não	Sim	Não
Albânia	AL	355	Não	Não	Sim	Não
Argélia	DZ	213	Não	Não	Sim	Não
Andorra	AD	376	Não	Não	Sim	Não
Anguila	AI	1-264	Não	Não	Sim	Não
Antígua e Barbuda	AG	1-268	Não	Não	Sim	Não
Argentina	AR	54	Sim	Não	Não	Não
Armênia	AM	374	Não	Não	Sim	Não
Aruba	AW	297	Não	Não	Sim	Não
Austrália	AU	61	Não	Sim	É necessário registro <sup>1</sup>	Sim
Áustria	AT	43	Sim	Sim	Sim	Sim
Azerbaijão	AZ	994	Não	Não	Sim	Não
<b>B</b>						
Bahamas	BS	1-242	Não	Não	Não	Não

País ou região	Código ISO	Código de discagem	É compatível com códigos curtos	É compatível com códigos longos	Oferece suporte a IDs de remetente	Compatível com SMS bidirecional
Bahrein	BH	973	Não	Não	Sim	Não
Bangladesh	BD	880	Não	Não	Sim	Não
Barbados	BB	1-246	Não	Não	Sim	Não
Bielorrússia	BY	375	Não	Não	É necessário registro <sup>1</sup>	Não
Bélgica	BE	32	Não	Sim	Não	Sim
Belize	BZ	501	Não	Não	Sim	Não
Bermudas	BM	1-441	Não	Não	Sim	Não
Butão	BT	975	Não	Não	Sim	Não
Bolívia	BO	591	Não	Não	Sim	Não
Bósnia e Herzegovina	BA	387	Não	Não	Sim	Não
Botsuana	BW	267	Não	Não	Sim	Não
Brasil	BR	55	Sim	Não	Não	Sim
Brunei	BN	673	Não	Não	Sim	Não
Bulgária	BG	359	Sim	Não	Sim	Sim

País ou região	Código ISO	Código de discagem	É compatível com códigos curtos	É compatível com códigos longos	Oferece suporte a IDs de remetente	Compatível com SMS bidirecional
Burkina Faso	BF	226	Não	Não	Sim	Não
Burundi	BL	257	Não	Não	Sim	Não
<b>C</b>						
Camboja	KH	855	Não	Não	Sim	Não
Camarões	CM	237	Não	Não	Sim	Não
Canadá	CA	1	Sim	Sim	Não	Sim
Cabo Verde	CV	238	Não	Não	Sim	Não
Ilhas Cayman	KY	1-345	Não	Não	Não	Não
República Centro-Africana	CF	236	Não	Não	Sim	Não
Chade	TD	235	Não	Não	Sim	Não
Chile	CL	56	Sim	Sim	Não	Sim
China	CN	86	Sim	Não	N. <a href="#">2</a>	Sim
Colômbia	CO	57	Sim	Sim	Não	Sim
Comoros	KM	269	Não	Não	Sim	Não
Ilhas Cook	CK	682	Não	Não	Sim	Sim

País ou região	Código ISO	Código de discagem	É compatível com códigos curtos	É compatível com códigos longos	Oferece suporte a IDs de remetente	Compatível com SMS bidirecional
Costa Rica	CR	506	Não	Não	Não	Não
Croácia	HR	385	Não	Não	Sim	Não
Chipre	CY	357	Não	Não	Sim	Não
República Tcheca (Tchéquia)	CZ	420	Não	Sim	Sim	Sim
D						
República Democrática do Congo	CD	243	Não	Não	Sim	Não
Dinamarca	DK	45	Sim	Sim	Sim	Sim
Djibuti	DJ	253	Não	Não	Sim	Não
Dominica	DM	1-767	Não	Não	Sim	Não
República Dominicana	DO	1-809, 1-829, 1-849	Sim	Não	Não	Sim
E						
Equador	EC	593	Sim	Não	Não	Sim
Egito	EG	20	Sim	Não	É necessário registro <sup>1</sup>	Sim

País ou região	Código ISO	Código de discagem	É compatível com códigos curtos	É compatível com códigos longos	Oferece suporte a IDs de remetente	Compatível com SMS bidirecional
El Salvador	SV	503	Não	Não	Não	Não
Guiné Equatorial	GQ	240	Não	Não	Sim	Não
Eritreia	ER	291	Não	Não	Sim	Não
Estônia	EE	372	Não	Sim	Sim	Sim
Etiópia	ET	251	Não	Não	Sim	Não
F						
Ilhas Faroe	FO	298	Não	Não	Sim	Não
Fiji	FJ	679	Não	Não	Sim	Não
Finlândia	FI	358	Sim	Sim	Sim	Sim
França	FR	33	Sim	Não	Sim	Sim
Guiana Francesa	GF	594	Não	Não	Sim	Não
Polinésia Francesa	PF	689	Não	Não	Sim	Não
G						
Gabão	GA	241	Não	Não	Sim	Não
Gâmbia	GM	220	Não	Não	Sim	Não
Geórgia	GE	995	Não	Não	Sim	Não

País ou região	Código ISO	Código de discagem	É compatível com códigos curtos	É compatível com códigos longos	Oferece suporte a IDs de remetente	Compatível com SMS bidirecional
Alemanha	DE	49	Sim	Sim	Sim	Sim
Gana	GH	233	Não	Não	Sim	Não
Gibraltar	GI	350	Não	Não	Sim	Não
Grécia	GR	30	Não	Não	Sim	Não
Groenlândia	GL	299	Não	Não	Sim	Não
Granada	GD	1-473	Não	Não	Sim	Não
Guadalupe	GP	590	Não	Não	Sim	Não
Guam	GU	1-671	Não	Não	Não	Sim
Guatemala	GT	502	Não	Não	Não	Não
Guernsey	GG	44-1481	Não	Não	Sim	Não
Guiné	GN	224	Não	Não	Sim	Não
Guiné-Bissau	GW	245	Não	Não	Sim	N/D
Guiana	GY	592	Não	Não	Sim	Não
H						
Haiti	H	509	Não	Não	Sim	Não
Honduras	HN	504	Não	Não	Sim	Não
Hong Kong	HK	852	Não	Sim	Sim	Sim

País ou região	Código ISO	Código de discagem	É compatível com códigos curtos	É compatível com códigos longos	Oferece suporte a IDs de remetente	Compatível com SMS bidirecional
Hungria	HU	36	Não	Sim	Não	Sim
I						
Islândia	IS	354	Não	Não	Sim	Não
Índia	IN	91	Sim	Sim <sup>4</sup>	É necessário registro <sup>3</sup>	Sim
Indonésia	ID	62	Não	Não	Sim	Não
Iraque	IQ	964	Não	Não	Sim	Não
Irlanda	IE	353	Não	Sim	Sim	Sim
Ilha de Man	IM	44-1624	Não	Não	Sim	Não
Israel	IL	972	Não	Sim	Sim	Sim
Itália	IT	39	Sim	Sim	Sim	Sim
Costa do Marfim	CI	225	Não	Não	Sim	Não
J						
Jamaica	JM	1-876	Não	Não	Sim	Não
Japão	JP	81	Sim	Sim	Sim	Sim
Jérsei	JE	44-1534	Não	Sim	Sim	Sim

País ou região	Código ISO	Código de discagem	É compatível com códigos curtos	É compatível com códigos longos	Oferece suporte a IDs de remetente	Compatível com SMS bidirecional
Jordânia	JO	962	Não	Não	É necessário registro <sup>1</sup>	Não

## K

Cazaquistão	KZ	7	Não	Não	Sim	Não
Quênia	KE	254	Não	Não	Sim	Não
Kosovo	XK	383	Não	Não	Sim	Não
Kuwait	KW	965	Não	Não	É necessário registro <sup>1</sup>	Não
Quirguistão	KG	996	Não	Não	Sim	Não

## L

Laos	LA	856	Não	Não	Sim	Não
Letônia	LV	371	Não	Não	Sim	Não
Líbano	LB	961	Não	Não	Sim	Não
Lesoto	LS	266	Não	Não	Sim	Não
Libéria	LR	231	Não	Sim	Não	
Líbia	LY	218	Não	Não	Sim	Não

País ou região	Código ISO	Código de discagem	É compatível com códigos curtos	É compatível com códigos longos	Oferece suporte a IDs de remetente	Compatível com SMS bidirecional
Liechtenstein	LI	423	Não	Não	Sim	Não
Lituânia	LT	370	Não	Sim	Sim	Sim
Luxemburgo	LU	352	Não	Sim	Sim	Sim
M						
Macau	MO	853	Não	Não	Sim	Não
Macedônia	MK	389	Não	Não	Sim	Não
Madagascar	MG	261	Não	Não	Sim	Não
Malawi	MW	265	Não	Não	Sim	Não
Malásia	MY	60	Sim	Não	Não	Sim
Ilhas Maldivas	MV	960	Não	Não	Sim	Não
Mali	ML	223	Não	Não	Sim	Não
Malta	MT	356	Não	Não	Sim	Não
Ilhas Marshall	MH	692	Não	Não	Não	Não
Martinica	MQ	596	Não	Não	Sim	Não
Mauritânia	MR	222	Não	Não	Sim	Não

País ou região	Código ISO	Código de discagem	É compatível com códigos curtos	É compatível com códigos longos	Oferece suporte a IDs de remetente	Compatível com SMS bidirecional
Ilhas Maurício	MU	230	Não	Não	Sim	Não
Mayotte	YT	262	Não	Não	Sim	Não
México	MX	52	Sim	Não	Não	Sim
Micronésia (Estados Federados da)	FM	691	Não	Não	Não	Não
Moldávia	MD	373	Não	Não	Sim	Não
Mônaco	MC	377	Não	Não	Não	Não
Mongólia	MN	976	Não	Não	Sim	Não
Montenegro	ME	382	Não	Não	Sim	Não
Montserrat	MS	1-664	Não	Não	Sim	Não
Marrocos	MA	212	Sim	Não	Sim	Sim
Moçambique	MZ	258	Não	Não	Não	Não
Mianmar	MM	95	Não	Sim	Sim	Sim
N						
Namíbia	NA	264	Não	Não	Sim	Não

País ou região	Código ISO	Código de discagem	É compatível com códigos curtos	É compatível com códigos longos	Oferece suporte a IDs de remetente	Compatível com SMS bidirecional
Nepal	NP	977	Não	Não	Sim	Não
Holanda	NL	31	Sim	Sim	Sim	Sim
Antilhas Holandesas	AN	599	Não	Não	Sim	Não
Nova Caledônia	NC	687	Não	Não	Sim	Não
Nova Zelândia <sup>6</sup>	NZ	64	Sim	Não	Não	Sim
Nicarágua	NI	505	Não	Não	Não	Não
Níger	NE	227	Não	Não	Sim	Não
Nigéria	NG	234	Não	Não	Sim	Não
Niue	NU	683	Não	Não	Sim	Não
Noruega	NO	47	Não	Sim	Sim	Sim
O						
Omã	OM	968	Não	Não	Não	N/D
P						
Paquistão	PK	92	Não	Não	Sim	N/D
Palestina	PS	970	Não	Não	Sim	Não

País ou região	Código ISO	Código de discagem	É compatível com códigos curtos	É compatível com códigos longos	Oferece suporte a IDs de remetente	Compatível com SMS bidirecional
Panamá	PA	507	Não	Não	Sim	Não
Papua Nova Guiné	PG	675	Não	Não	Sim	Não
Paraguai	PY	595	Não	Não	Não	Não
Peru	PE	51	Sim	Não	Não	Sim
Filipinas	PH	63	Não	Sim	É necessário registro <sup>1</sup>	Sim
Polônia	PL	48	Não	Sim	Sim	Sim
Portugal	PT	351	Não	Sim	Sim	Sim
Porto Rico	PR	1-797, 1-939	Não	Não	Não	Sim
Q						
Catar	QA	974	Não	Não	É necessário registro <sup>1</sup>	Não
R						
República do Congo	CG	242	Não	Não	Não	Não

País ou região	Código ISO	Código de discagem	É compatível com códigos curtos	É compatível com códigos longos	Oferece suporte a IDs de remetente	Compatível com SMS bidirecional
Reunião (França)	RE	262	Não	Não	Sim	Não
Romênia	RO	40	Não	Sim	Sim	Sim
Rússia	RU	7	Sim	Não	É necessário registro <sup>1</sup>	Sim
Ruanda	RW	250	Não	Não	Sim	Não
S						
São Cristóvão e Nevis	KN	1-869	Não	Não	Não	Não
Santa Lúcia	LC	1-758	Não	Não	Não	Não
Samoa	WS	685	Não	Não	Sim	Não
São Marinho	SM	378	Não	Não	Sim	Não
São Tomé e Príncipe	ST	239	Não	Não	Sim	Não
Arábia Saudita	SA	966	Não	Sim <sup>4</sup>	É necessário registro <sup>1</sup>	Não
Senegal	SN	221	Não	Não	Sim	Não

País ou região	Código ISO	Código de discagem	É compatível com códigos curtos	É compatível com códigos longos	Oferece suporte a IDs de remetente	Compatível com SMS bidirecional
Sérvia	RS	381	Não	Não	Sim	Não
Seichelles	SC	248	Não	Não	Sim	Não
Serra Leoa	SL	232	Não	Não	Sim	Não
Singapura	SG	65	Sim	Sim	Sim <sup>5</sup>	Sim
Eslováquia	SK	421	Não	Sim	Sim	Não
Eslovênia	SI	386	Não	Não	Sim	Não
Ilhas Salomão	SB	677	Não	Não	Sim	Não
Somália	SO	252	Não	Não	Sim	Não
África do Sul	ZA	27	Sim	Sim	Não	Sim
Coreia do Sul	KR	82	Não	Não	Não	Não
Sudão do Sul	SS	211	Não	Não	Sim	Não
Espanha	ES	34	Sim	Sim	Sim	Sim
Sri Lanka	LK	94	Não	Não	É necessário registro <sup>1</sup>	Não
Suriname	SR	597	Não	Não	Sim	Não

País ou região	Código ISO	Código de discagem	É compatível com códigos curtos	É compatível com códigos longos	Oferece suporte a IDs de remetente	Compatível com SMS bidirecional
Suazilândia	SZ	268	Não	Não	Sim	Não
Suécia	SE	46	Sim	Sim	Sim	Sim
Suíça	CH	41	Não	Sim	Sim	Sim
T						
Taiwan	TW	886	Não	Sim	Não	Sim
Tajiquistão	TJ	992	Não	Não	Sim	Não
Tanzânia	TX	255	Não	Não	Sim	Não
Tailândia	TH	66	Não	Sim	É necessário registro <sup>1</sup>	Sim
Timor-Leste	TL	670	Não	Não	Sim	Não
Togo	TG	228	Não	Não	Sim	Não
Tonga	TO	676	Não	Não	Sim	Não
Trinidad e Tobago	TT	1-868	Não	Não	Sim	Não
Tunísia	TN	216	Não	Não	Sim	Não
Turquia	TR	90	Não	Não	É necessário registro <sup>1</sup>	Não

País ou região	Código ISO	Código de discagem	É compatível com códigos curtos	É compatível com códigos longos	Oferece suporte a IDs de remetente	Compatível com SMS bidirecional
Turcomenistão	TM	993	Não	Não	Não	Não
Ilhas Turcas e Caicos	TC	1-649	Não	Não	Sim	Não
Tuvalu	TC	688	Não	Não	Sim	Não
U						
Uganda	UG	256	Não	Não	Sim	Não
Ucrânia	UA	380	Não	Sim	Sim	Sim
Emirados Árabes Unidos (EAU)	AE	971	Sim	Sim	É necessário registro <sup>1</sup>	Sim
Reino Unido	GB	44	Sim	Sim	Sim	Sim
Estados Unidos	US	1	Sim	Sim	Não	Sim
Uruguai	UY	598	Sim	Não	Não	Sim
Uzbequistão	UZ	998	Não	Não	Sim	Não
V						
Vanuatu	VU	678	Não	Não	Sim	Não

País ou região	Código ISO	Código de discagem	É compatível com códigos curtos	É compatível com códigos longos	Oferece suporte a IDs de remetente	Compatível com SMS bidirecional
Venezuela	VE	58	Não	Não	Não	Não
Vietnã	VN	84	Não	Não	É necessário registro <sup>1</sup>	Não
Ilhas Virgens Britânicas	VG	1-284	Não	Não	Sim	Não
Ilhas Virgens, US	VI	1-340	Não	Não	Não	Sim
W						
X						
Y						
Iêmen	YE	967	Não	Não	Sim	Não
Z						
Zâmbia	ZM	260	Não	Não	Sim	Não
Zimbábue	ZW	263	Não	Não	Sim	Não

## Observações

- Os remetentes devem usar um ID de remetente alfabético pré-registrado. Para solicitar uma ID de remetente de AWS Support, consulte [Solicitar IDs do remetente para sistema de mensagens](#)

[por SMS com o Amazon SNS](#). Alguns países exigem que os remetentes atendam a requisitos específicos ou cumpram determinadas restrições para obter aprovação. Nesses casos, AWS Support pode entrar em contato com você para obter informações adicionais depois de enviar sua solicitação de ID de remetente.

2.

Os remetentes devem usar um modelo pré-registrado para cada tipo de mensagem que planejam enviar. Se um remetente não atender a esse requisito, as mensagens dele serão bloqueadas. Para registrar um modelo, abra um caso de SMS do Amazon SNS com o AWS Support. Ao criar o caso, forneça as mesmas informações que você forneceria para solicitar um ID de remetente. Para ter mais informações, consulte [Solicitar IDs do remetente para sistema de mensagens por SMS com o Amazon SNS](#). Alguns países exigem que os remetentes atendam a requisitos adicionais específicos ou cumpram determinadas restrições para obter aprovação. Nesses casos, AWS Support pode solicitar informações adicionais.

 Note

Para enviar mensagens para a China, você deve primeiro registrar seus modelos AWS Support para aprovação.

3.

Os remetentes devem usar um ID de remetente alfabético pré-registrado. Etapas adicionais de registro são necessárias. Para ter mais informações, consulte [Requisitos para registro de ID de remetente da Índia](#).

4.

Os códigos longos nesses países comportam apenas mensagens de entrada. Em outras palavras, você não pode usar esses códigos longos para enviar mensagens para seus destinatários, mas pode usá-los para receber mensagens de seus destinatários. Esses códigos longos são uma maneira útil de permitir que seus destinatários recusem mensagens se você as enviar usando um ID de remetente alfabético, porque os IDs de remetente comportam apenas mensagens de saída.

5.

O Amazon SNS pode enviar tráfego de SMS para Singapura usando um ID do remetente registrado no Registro de ID de Remetente de SMS de Singapura (SSIR), um registro criado pela [Autoridade de Desenvolvimento de Mídia de Informação e Comunicação \(IMDA\)](#) de Singapura. Para obter mais informações sobre os requisitos para usar um ID de remetente de Singapura, consulte [Requisitos para registro de ID de remetente de Singapura](#).

Você também pode enviar tráfego de SMS em Singapura usando IDs de remetente não registrados ou tipos alternativos de identidade de origem, como códigos curtos ou longos.

6. Sem um código simplificado dedicado, o Amazon SNS ainda tenta enviar mensagens aos destinatários da Nova Zelândia usando um conjunto compartilhado de códigos simplificados. Devido às restrições da operadora local em relação a números compartilhados, a entrega desses números é realizada com base no melhor esforço. Portanto, o Amazon SNS recomenda fortemente a aquisição de um código simplificado dedicado para todo tráfego enviado à Nova Zelândia. As mensagens que contêm URLs devem ser listadas como permitidas por meio do processo de código simplificado dedicado. Para obter mais informações sobre como comprar um código simplificado, consulte [Solicitar códigos curtos dedicados para o sistema de mensagens por SMS com o Amazon SNS](#).

## Práticas recomendadas para SMS

Os usuários de telefone celular tendem a ter uma tolerância muito baixa com mensagens SMS não solicitadas. As taxas de resposta para campanhas SMS não solicitadas quase sempre serão baixas e, portanto, o retorno do investimento será fraco.

Além disso, as operadoras de celular continuamente fazem auditoria em remetentes de SMS em massa. Elas limitam ou bloqueiam mensagens de números que elas determinam que estão enviando mensagens não solicitadas.

O envio de conteúdo não solicitado também é uma violação da [Política de uso aceitável da AWS](#). A equipe do Amazon SNS audita campanhas de SMS de maneira rotineira e poderá limitar ou bloquear a sua capacidade de enviar mensagens se você estiver enviando mensagens não solicitadas.

Por fim, em muitos países, regiões e jurisdições, há graves penalidades para o envio de mensagens SMS não solicitadas. Por exemplo, nos Estados Unidos, o Telephone Consumer Protection Act (TCPA) afirma que os consumidores podem ganhar indenizações de US\$ 500 a US\$ 1.500 (pagos pelo remetente) para cada mensagem não solicitada que recebem.

Esta seção descreve várias práticas recomendadas que podem ajudar você a melhorar seu envolvimento com os clientes e evitar multas pesadas. No entanto, observe que esta seção não contém orientação jurídica. Sempre consulte seu advogado para obter orientação jurídica.

### Tópicos

- [Cumprir as leis, os regulamentos e os requisitos da operadora](#)
- [Obter permissão](#)
- [Não enviar para listas antigas](#)

- [Fazer auditoria em suas listas de clientes](#)
- [Manter registros](#)
- [Torne suas mensagens claras, sinceras e concisas](#)
- [Responder de maneira apropriada](#)
- [Ajustar seu envio com base no envolvimento](#)
- [Enviar em momentos adequados](#)
- [Evitar a fadiga entre canais](#)
- [Usar códigos curtos dedicados](#)
- [Verificar seus números de telefone de destino](#)
- [Ter a redundância em mente ao projetar](#)
- [Limites e restrições de SMS](#)
- [Gerenciamento de palavras-chave de cancelamento](#)
- [CreatePool](#)
- [PutKeyword](#)
- [Gerenciar configurações de números](#)
- [Limites de caracteres de SMS no Amazon SNS](#)

## Cumprir as leis, os regulamentos e os requisitos da operadora

Você pode enfrentar multas e penalidades significativas se violar as leis e os regulamentos dos lugares onde seus clientes residem. Por esse motivo, é importante compreender as leis relacionadas a mensagens SMS em cada país ou região em que você faz negócios.

A lista a seguir inclui links para as principais leis que se aplicam às comunicações SMS em mercados importantes em todo o mundo.

- Estados Unidos: o Telephone Consumer Protection Act de 1991, também conhecido como TCPA, aplica-se a certos tipos de mensagens SMS. Para obter mais informações, consulte [regras e regulamentos](#) no site da Federal Communications Commission.
- Reino Unido: as Privacy and Electronic Communications (EC Directive) Regulations 2003, também conhecidas como PECR, aplicam-se a certos tipos de mensagens SMS. Para obter mais informações, consulte [O que são PECR?](#) no site do Information Commissioner's Office do Reino Unido.

- União Europeia: a Privacy and Electronic Communications Directive 2002, também conhecida como ePrivacy Directive, aplica-se a alguns tipos de mensagens SMS. Para obter mais informações, consulte o [texto completo da lei](#) no site Europa.eu.
- Canadá: o Fighting Internet and Wireless Spam Act, mais comumente conhecido como Anti-Spam Law ou CASL do Canadá, aplica-se a certos tipos de mensagens SMS. Para obter mais informações, consulte o [texto completo da lei](#) no site do Parlamento do Canadá.
- Japão: o Act on Regulation of Transmission of Specific Electronic Mail pode ser aplicável a determinados tipos de mensagens SMS. Para obter mais informações, consulte [Contra medidas contra spam do Japão](#) no site do Ministry of Internal Affairs and Communications japonês.

Como remetente, essas leis podem se aplicar a você mesmo que sua empresa ou organização não esteja sediada em um desses países. Algumas das leis nesta lista foram originalmente criadas para abordar e-mails ou telefonemas não solicitados, mas foram interpretados ou expandidos para que fossem aplicados às mensagens SMS. Outros países e regiões podem ter suas próprias leis relacionadas à transmissão de mensagens SMS. Consulte um advogado em cada país ou região em que seus clientes estão localizados para obter orientação jurídica.

Em muitos países, as operadoras locais têm autoridade para determinar que tipo de tráfego flui em suas redes. Isso significa que elas podem impor restrições ao conteúdo de SMS que exceda os requisitos mínimos das leis locais.

## Obter permissão

Nunca envie mensagens a destinatários que não tenham solicitado explicitamente o recebimento dos tipos específicos de mensagens que você planeja enviar. Não compartilhe listas de aceitação, mesmo entre organizações da mesma empresa.

Se os destinatários puderem se cadastrar para receber suas mensagens usando um formulário on-line, adicione sistemas para evitar que scripts automatizados inscrevam pessoas sem o conhecimento delas. Você também deve limitar o número de vezes que um usuário pode enviar um número de telefone em uma única sessão.

Ao receber uma solicitação de aceitação de SMS, envie uma mensagem ao destinatário pedindo para confirmar que ele deseja receber mensagens de você. Não envie a esse destinatário mensagens adicionais enquanto ele não confirmar a assinatura. Uma mensagem de confirmação de inscrição pode ser parecida com o exemplo a seguir:

Text YES to join ExampleCorp alerts. 2 msgs/month. Msg & data rates may apply. Reply HELP for help, STOP to cancel.

Mantenha registros que incluem a data, a hora e a origem de cada solicitação de inclusão e confirmação de inscrição. Isso pode ser útil se uma operadora ou uma agência reguladora solicitar, e também pode ajudar você a realizar auditorias de rotina de sua lista de clientes.

### Fluxo de trabalho de aceitação

Em alguns casos (como o registro de código curto ou de chamada gratuita dos EUA), as operadoras de celular exigem que você forneça modelos ou capturas de tela de todo o seu fluxo de trabalho de aceitação. Os modelos ou as capturas de tela devem ser bem semelhantes ao fluxo de trabalho de aceitação que seus destinatários concluirão.

Seus modelos ou capturas de tela devem incluir todas as divulgações necessárias listadas abaixo para manter o mais alto nível de conformidade.

### Divulgações obrigatórias

- Uma descrição do caso de uso de mensagens que você enviará por meio de seu programa.
- A frase “Taxas de mensagens e dados podem ser aplicadas”.
- Uma indicação da frequência com que os destinatários receberão suas mensagens. Por exemplo, um programa de mensagens recorrentes pode indicar “uma mensagem por semana”. Um caso de uso de senha de uso único ou autenticação multifator pode indicar “a frequência da mensagem varia” ou “uma única mensagem por tentativa de login”.
- Links para seus documentos de termos e condições e política de privacidade.

### Motivos comuns de rejeição de aceitação não compatíveis

- Se o nome da empresa fornecido não corresponder ao fornecido no modelo ou na captura de tela. Qualquer relação não óbvia deve ser explicada na descrição do fluxo de trabalho de aceitação.
- Se parecer que uma mensagem será enviada ao destinatário, mas nenhum consentimento foi obtido explicitamente antes disso. O consentimento explícito é um requisito de todas as mensagens.
- Se parecer que é necessário receber uma mensagem de texto para cadastramento em um serviço. Isso não será compatível se o fluxo de trabalho não fornecer nenhuma alternativa ao recebimento de uma mensagem de aceitação em outro formato, como e-mail ou chamada de voz.

- Se o teor da aceitação for apresentado inteiramente nos termos de serviço. As divulgações devem sempre ser apresentadas ao destinatário no momento da aceitação, em vez de armazenadas em um documento de política vinculado.
- Se um cliente consentir em receber um tipo de mensagem sua e você enviar a ele outros tipos de mensagem de texto. Por exemplo, ele consente em receber senhas de uso único, mas também recebe mensagens sobre enquetes e pesquisas.
- Se as divulgações exigidas (listadas acima) não forem apresentadas aos destinatários.

O exemplo a seguir está em conformidade com os requisitos das operadoras de celular para um caso de uso de autenticação multifator.

1. User provides basic account information.

2. User decides whether to enable MFA.

3. If MFA enabled, user chooses how to receive MFA token.



4. If user chooses to receive MFA token by text, send a token.

5. User enters MFA token to verify phone number.

## Modelo de caso de uso da autenticação multifator

Ele contém texto e imagens finalizados e mostra todo o fluxo de aceitação, complementado por anotações. No fluxo de aceitação, o cliente precisa tomar medidas distintas e intencionais para

fornecer seu consentimento para receber mensagens de texto e todas as divulgações necessárias estão incluídas.

### Outros tipos de fluxo de trabalho de aceitação

As operadoras de celular também aceitarão fluxos de trabalho de aceitação fora de aplicações e sites, como aceitação verbal ou por escrito, se estiverem em conformidade com o descrito acima. Um fluxo de trabalho de aceitação compatível e um script verbal ou por escrito obterão o consentimento explícito do destinatário para receber um tipo específico de mensagem. São exemplos o script verbal que um agente de suporte usa para obter consentimento antes de gravar em um banco de dados de serviços ou um número de telefone listado em um folheto promocional. Para fornecer um modelo desses tipos de fluxo de trabalho de aceitação, você pode fornecer uma captura de tela de seu script de aceitação, material de marketing ou banco de dados onde os números são coletados. As operadoras de celular poderão ter perguntas adicionais sobre esses casos de uso se uma aceitação não estiver clara ou se o caso de uso exceder determinados volumes.

### Não enviar para listas antigas

As pessoas mudam de número de telefone com frequência. Um número de telefone pelo qual você obteve consentimento para entrar em contato há dois anos pode pertencer a outra pessoa hoje. Não use listas antigas de números de telefone para um novo programa de mensagens; se você fizer isso, é provável que algumas mensagens sejam malsucedidas porque o número não está mais funcionando e algumas pessoas optam por não aceitar as mensagens porque não se lembram de ter dado consentimento.

### Fazer auditoria em suas listas de clientes

Se você envia campanhas de SMS recorrentes, faça uma auditoria em suas listas de cliente regularmente. Fazer uma auditoria nas listas de clientes garante que somente os clientes interessados em receber as mensagens são aqueles que as recebem.

Ao fazer uma auditoria em sua lista, envie a cada cliente incluído uma mensagem que lembre a ele que está inscrito e ofereça informações sobre o cancelamento da inscrição. Uma mensagem de lembrete pode ser parecida com o exemplo a seguir:

```
You're subscribed to ExampleCorp alerts. Msg & data rates may apply. Reply  
HELP for help, STOP to unsubscribe.
```

## Manter registros

Mantenha registros que mostram quando cada cliente solicitou o recebimento de mensagens SMS e quais mensagens você enviou para cada cliente. Muitos países e regiões do mundo exigem remetentes de SMS para manter esses registros de uma forma que possam ser facilmente recuperados. Operadoras de celular também podem solicitar essas informações a qualquer momento. As informações exatas que você deve fornecer variam de acordo com o país ou região. Para obter mais informações sobre os requisitos para manter registros, analise os regulamentos sobre o envio de mensagens SMS comerciais em cada país ou região dos clientes.

Às vezes, uma operadora ou agência reguladora pedem para fornecermos um comprovante de que um cliente optou por receber mensagens de você. Nesses casos, o AWS Support entra em contato com você com uma lista das informações que a operadora ou agência exigem. Se você não conseguir fornecer as informações necessárias, poderemos pausar a sua capacidade de enviar outras mensagens SMS.

## Torne suas mensagens claras, sinceras e concisas

O SMS é um meio que tem especificidades. O limite de 160 caracteres por mensagem determina que suas mensagens sejam concisas. Técnicas que você pode usar em outros canais de comunicação, como e-mail, podem não se aplicar ao canal de SMS e até parecer desonestas ou enganosas quando usadas com mensagens SMS. Se o conteúdo de suas mensagens não estiver de acordo com as práticas recomendadas, os destinatários poderão ignorá-las; na pior das hipóteses, as operadoras de celular poderão identificar suas mensagens como spam e bloquear futuras mensagens de seu número de telefone.

Esta seção fornece algumas dicas e ideias para criar um corpo de mensagem SMS eficaz.

### Identificar-se como remetente

Seus destinatários devem saber imediatamente que uma mensagem é sua. Os remetentes que seguem essa prática recomendada incluem um nome de identificação (“nome do programa”) no início de cada mensagem.

Não elabore mensagens deste tipo:

```
Your account has been accessed from a new device. Reply Y to confirm.
```

Em vez disso, tente:

ExampleCorp Financial Alerts: You have logged in to your account from a new device. Reply Y to confirm, or STOP to opt-out.

Não tentar fazer com que a mensagem pareça uma mensagem particular

Alguns profissionais de marketing ficam tentados a dar um toque pessoal às mensagens SMS fazendo com que pareçam vir de uma pessoa. No entanto, essa técnica pode fazer sua mensagem parecer uma tentativa de phishing.

Não elabore mensagens deste tipo:

Hi, this is Jane. Did you know that you can save up to 50% at Example.com? Click here for more info: <https://www.example.com>.

Em vez disso, tente:

ExampleCorp Offers: Save 25-50% on sale items at Example.com. Click here to browse the sale: <https://www.example.com>. Text STOP to opt-out.

Ter cuidado ao falar sobre dinheiro

Os golpistas geralmente se aproveitam do desejo das pessoas de economizar e receber dinheiro. Evite que as ofertas pareçam muito boas para serem verdadeiras. Não use a sedução de dinheiro para enganar as pessoas. Não use símbolos monetários para indicar dinheiro.

Não elabore mensagens deste tipo:

Save big \$\$\$ on your next car repair by going to <https://www.example.com>.

Em vez disso, tente:

ExampleCorp Offers: Your ExampleCorp insurance policy gets you discounts at 2300+ repair shops nationwide. More info at <https://www.example.com>. Text STOP to opt-out.

## Usar somente os caracteres necessários

Muitas vezes, as empresas tendem a proteger suas marcas registradas incluindo símbolos como ™ ou ® nas mensagens. No entanto, esses símbolos não fazem parte do conjunto padrão de caracteres (conhecido como alfabeto GSM) que pode ser incluído em uma mensagem SMS de 160 caracteres. Quando você envia uma mensagem com esses caracteres, ela é enviada automaticamente usando um sistema de codificação de caracteres diferente, que aceita apenas 70 caracteres por parte da mensagem. Por isso, sua mensagem pode ser dividida em várias partes. Como há cobrança por cada parte da mensagem enviada, isso pode ter um custo superior ao que você espera para enviar a mensagem completa. Além disso, os destinatários podem receber várias mensagens sequenciais de você em vez de uma única mensagem. Para obter mais informações sobre a codificação de caracteres do SMS, consulte [Limites de caracteres de SMS no Amazon SNS](#).

Não elabore mensagens deste tipo:

```
ExampleCorp Alerts: Save 20% when you buy a new ExampleCorp Widget® at
example.com and use the promo code WIDGET.
```

Em vez disso, tente:

```
ExampleCorp Alerts: Save 20% when you buy a new ExampleCorp Widget(R) at
example.com and use the promo code WIDGET.
```

### Note

Os dois exemplos anteriores são quase idênticos, mas o primeiro exemplo contém um símbolo de marca registrada (®), o qual não faz parte do alfabeto GSM. Por isso, o primeiro exemplo é enviado como duas partes da mensagem, enquanto o segundo é enviado como uma parte da mensagem.

## Usar links válidos e seguros

Se a sua mensagem incluir links, confira-os para verificar se eles funcionam. Teste os links em um dispositivo fora da rede corporativa para garantir que eles sejam resolvidos corretamente. Devido ao limite de 160 caracteres das mensagens SMS, URLs muito longos podem ser divididos em várias mensagens. Você precisa usar domínios de redirecionamento para fornecer URLs curtos. No entanto, você não deve usar serviços gratuitos de encurtamento de links, como [tinyurl.com](#) ou

bitly.com, porque as operadoras tendem a filtrar mensagens que incluem links nesses domínios. No entanto, você pode usar serviços pagos de encurtamento de links, desde que os links apontem para um domínio dedicado ao uso exclusivo de sua empresa ou organização.

Não elabore mensagens deste tipo:

```
Go to https://tinyurl.com/4585y8mr today for a special offer!
```

Em vez disso, tente:

```
ExampleCorp Offers: Today only, get an exclusive deal on an ExampleCorp Widget. See https://a.co/cFKmaRG for more info. Text STOP to opt-out.
```

Limitar o número de abreviações utilizadas

A limitação de 160 caracteres do canal de SMS leva alguns remetentes a acreditar que precisam usar abreviações extensivamente em suas mensagens. No entanto, o uso excessivo de abreviações pode parecer pouco profissional para muitos leitores e fazer com que alguns usuários denunciem sua mensagem como spam. É plenamente possível escrever uma mensagem coerente sem usar um número excessivo de abreviações.

Não elabore mensagens deste tipo:

```
Get a gr8 deal on ExampleCorp widgets when u buy a 4-pack 2day.
```

Em vez disso, tente:

```
ExampleCorp Alerts: Today only—an exclusive deal on ExampleCorp Widgets at example.com. Text STOP to opt-out.
```

## Responder de maneira apropriada

Quando um destinatário responde suas mensagens, responda com informações úteis. Por exemplo, quando um cliente responder a uma de suas mensagens com a palavra-chave "AJUDA", envie informações sobre o programa no qual ele está inscrito, o número de mensagens que você vai enviar por mês e as maneiras com as quais ele pode entrar em contato com você para obter mais informações. Uma resposta a uma mensagem de AJUDA pode ser parecida com o exemplo a seguir:

HELP: ExampleCorp alerts: email help@example.com or call 425-555-0199. 2 msgs/month. Msg & data rates may apply. Reply STOP to cancel.

Quando um cliente responder com a palavra-chave "PARAR", explique que ele não receberá mais mensagens. Uma resposta a uma mensagem de PARAR pode ser parecida com o exemplo a seguir:

You're unsubscribed from ExampleCorp alerts. No more messages will be sent. Reply HELP, email help@example.com, or call 425-555-0199 for more info.

## Ajustar seu envio com base no envolvimento

As prioridades de seus clientes podem mudar ao longo do tempo. Se os clientes não acham mais suas mensagens úteis, eles podem querer cancelar a inscrição para suas mensagens ou até mesmo informar suas mensagens como não solicitadas. Por esses motivos, é importante que você ajuste suas práticas de envio com base no envolvimento do cliente.

Você precisa ajustar a frequência de suas mensagens para clientes que raramente interagem com elas. Por exemplo, se você envia mensagens semanais para clientes envolvidos, pode criar uma compilação mensal separada para os clientes com menos envolvimento.

Por fim, remova das suas listas de clientes aqueles que não têm nenhum envolvimento. Essa etapa impede que os clientes fiquem frustrados com suas mensagens. Isso também gera economia e ajuda a proteger sua reputação como remetente.

## Enviar em momentos adequados

Somente envie mensagens durante o horário comercial normal. Se você envia mensagens na hora do jantar ou no meio da noite, há uma boa chance de que seus clientes cancelarão a inscrição de suas listas para não serem mais perturbados. Além disso, não faz sentido enviar mensagens SMS quando os clientes não podem responder a elas imediatamente.

Se você enviar campanhas ou jornadas para públicos muito grandes, confira as taxas de throughput de seus números de originação. Divida o número de destinatários pela taxa de throughput para determinar quanto tempo levará para enviar mensagens a todos os destinatários.

## Evitar a fadiga entre canais

Se você usa vários canais de comunicação (como e-mail, SMS e mensagens push) em suas campanhas, não envie a mesma mensagem em cada canal. Quando você envia a mesma mensagem ao mesmo tempo em mais de um canal, seus clientes provavelmente percebem o comportamento de envio como irritante em vez de útil.

## Usar códigos curtos dedicados

Se você usa códigos simplificados, mantenha um separado para cada marca e cada tipo de mensagem. Por exemplo, se a sua empresa tem duas marcas, use um código simplificado separado para cada uma. Da mesma forma, se você envia mensagens promocionais e transacionais, use um código simplificado separado para cada tipo de mensagem. Para saber mais sobre como solicitar códigos simplificados, consulte [Solicitar códigos curtos dedicados para o sistema de mensagens por SMS com o Amazon SNS](#).

## Verificar seus números de telefone de destino

Ao enviar mensagens SMS pelo Amazon SNS, cada parte da mensagem enviada é cobrada. O preço pago por parte da mensagem varia de acordo com o país ou a região do destinatário. Para obter mais informações sobre preços do SMS, consulte [Definição de preço do Amazon SMS](#).

Quando o Amazon SNS aceita uma solicitação para enviar uma mensagem SMS (como resultado de uma chamada para a API [SendMessage](#) ou do lançamento de uma campanha ou jornada), há cobrança pelo envio dessa mensagem. Essa afirmação se aplica mesmo que o destinatário pretendido não receba realmente a mensagem. Por exemplo, se o número de telefone do destinatário não estiver mais funcionando ou se o número para o qual você enviou a mensagem não for um número de celular válido, mesmo assim haverá cobrança pelo envio da mensagem.

O Amazon SNS aceita solicitações válidas para enviar mensagens SMS e tenta entregá-las. Por esse motivo, você deve validar se os números de telefone para os quais você envia mensagens são números de celular válidos. Você pode usar o serviço de validação de número de telefone do Amazon SNS para determinar se um número de telefone é válido e qual é o tipo de número (como celular, telefone fixo ou VoIP). Para obter mais informações, consulte [Validating phone numbers in Amazon Pinpoint](#) (“Validar números de telefone no Amazon Pinpoint”) no Guia do desenvolvedor do Amazon Pinpoint.

## Ter a redundância em mente ao projetar

Para programas de mensagens essenciais, recomendamos que você configure o Amazon SNS em mais de uma Região da AWS. O Amazon SNS está disponível em diversas Regiões da AWS. Para obter uma lista de regiões onde o Amazon SNS está disponível, consulte a [Referência geral da AWS](#).

Os números de telefone que você usa para mensagens SMS, inclusive códigos curtos, códigos longos, números gratuitos e números 10DLC, não podem ser replicados entre Regiões da AWS. Por isso, para usar o Amazon SNS em várias regiões, você deve solicitar números de telefone separados em cada região em que deseja usar o Amazon SNS. Por exemplo, se você usar um código curto

para enviar mensagens de texto a destinatários nos Estados Unidos, precisará solicitar códigos curtos separados em cada Região da AWS que pretenda usar.

Em alguns países, você também pode usar vários tipos de número de telefone para aumentar a redundância. Por exemplo, nos Estados Unidos, você pode solicitar códigos curtos, números 10DLC e números gratuitos. Cada um desses tipos de número de telefone segue um caminho diferente até o destinatário. Ter vários tipos de número de telefone disponíveis, seja na mesma Região da AWS ou distribuídos por várias Regiões da AWS, oferece uma camada adicional de redundância, que pode ajudar a melhorar a resiliência.

## Limites e restrições de SMS

Para ver os limites e restrições de SMS, consulte [SMS limits and restrictions in Amazon Pinpoint](#) (Limites e restrições de SMS no Amazon Pinpoint) no Guia do usuário do Amazon Pinpoint.

## Gerenciamento de palavras-chave de cancelamento

Os destinatários de SMS podem usar seus dispositivos para cancelar o recebimento de mensagens respondendo com uma palavra-chave. Para obter mais informações, consulte [Cancelar recebimento de mensagens SMS](#).

## CreatePool

Use a ação da API `CreatePool` para criar um grupo e associar uma identidade de origem especificada ao grupo. Para obter mais informações, consulte [CreatePool](#) na API de mensagens SMS e de voz do Amazon Pinpoint.

## PutKeyword

Use a ação da API `PutKeyword` para criar ou atualizar uma configuração de uma palavra-chave em um grupo ou número de telefone de origem. Para obter mais informações, consulte [PutKeyword](#) na API de mensagens SMS e de voz do Amazon Pinpoint.

## Gerenciar configurações de números

Você pode usar as opções na seção `Number settings` (Configurações de números) da página `SMS and voice settings` (Configurações de mensagens SMS e de voz) para gerenciar as configurações dos códigos longos e simplificados dedicados que você solicitou ao AWS Support e atribuiu à sua conta. Para obter mais informações, consulte [Managing number settings](#) (Gerenciamento das configurações de número) no Guia do usuário do Amazon Pinpoint.

## Limites de caracteres de SMS no Amazon SNS

Uma mensagem SMS pode ter até 140 bytes de informação. O número de caracteres que você pode incluir em uma única mensagem SMS depende do tipo de caracteres contidos na mensagem.

Se a mensagem tiver somente [caracteres do conjunto de caracteres GSM 03.38](#), também conhecido como alfabeto GSM de 7 bits, ela poderá ter até 160 caracteres. Se a mensagem tiver caracteres que não façam parte do conjunto de caracteres GSM 03.38, ela poderá ter até 70 caracteres. Ao enviar uma mensagem SMS, o Amazon SNS determina automaticamente a codificação mais eficiente a ser usada.

Quando uma mensagem tiver mais do que o número máximo de caracteres, ela será dividida em várias partes. Quando as mensagens são divididas em várias partes, cada parte contém informações adicionais sobre a parte da mensagem que a precede. Quando o dispositivo do destinatário recebe partes da mensagem que estão separadas dessa forma, ele usa essas informações adicionais para garantir que todas as partes da mensagem sejam exibidas na ordem correta. Dependendo da operadora de celular e do dispositivo do destinatário, várias mensagens podem ser exibidas como uma única mensagem ou como uma sequência de mensagens separadas. Como resultado, o número de caracteres em cada parte da mensagem é reduzido a 153 (para mensagens que contenham somente caracteres GSM 03.38) ou 67 (para mensagens que contenham outros caracteres). É possível estimar quantas partes de mensagem sua mensagem contém antes de enviá-la usando ferramentas de calculadora de comprimento de SMS, muitas das quais estão disponíveis online. O tamanho máximo compatível para qualquer mensagem é de 1,6 mil caracteres GSM ou 630 caracteres não GSM. Para obter mais informações sobre throughput e tamanho de mensagens, consulte [Limites de caracteres de SMS no Amazon Pinpoint](#) no Guia do usuário do Amazon Pinpoint.

Para exibir o número de partes da mensagem para cada mensagem enviada, primeiro é necessário habilitar as [configurações de streaming de eventos](#). Ao fazer isso, o Amazon SNS produzirá um evento `_SMS.SUCCESS` quando a mensagem for entregue à operadora de telefonia móvel do destinatário. O registro do evento `_SMS.SUCCESS` contém um atributo chamado `attributes.number_of_message_parts`. Esse atributo especifica o número de partes da mensagem que a mensagem continha.

### Important

Quando enviar uma mensagem que contém mais de uma parte, você será cobrado pelo número de partes contidas na mensagem.

## Conjunto de caracteres GSM 03.38

A tabela a seguir indica todos os caracteres presentes no conjunto de caracteres GSM 03.38. Se você enviar uma mensagem que inclui apenas os caracteres mostrados na tabela a seguir, a mensagem poderá ter até 160 caracteres.

Caracteres padrão GSM 03.38												
A	B	C	D	E	F	G	H	I	J	K	L	M
N	O	P	Q	R	S	T	U	V	W	X	Y	Z
a	b	c	d	e	f	g	h	i	j	k	l	m
n	o	p	q	r	s	t	u	v	w	x	y	z
à	Å	å	Ä	ä	Ç	É	é	è	ì	Ñ	ñ	ò
Ø	ø	Ö	ö	ù	Ü	ü	Æ	æ	ß	0	1	2
3	4	5	6	7	8	9	&	*	@	:	,	¤
\$	=	!	>	#	-		¿	(	<	%	.	+
£	?	"	)	§	;	'	/	_	¥	Δ	Φ	Γ
Λ	Ω	Π	Ψ	Σ	Θ	Ξ						

O conjunto de caracteres GSM 03.38 inclui vários símbolos além dos mostrados na tabela anterior. No entanto, cada um desses caracteres é contado como dois caracteres, pois também inclui um caractere de escape invisível:

- ^
- {
- }
- \
- [
- ]

- ~
- |
- €

Por fim, o conjunto de caracteres GSM 03.38 também inclui os seguintes caracteres não impressos:

- Um caractere de espaço.
- Um controle de avanço de linha, que significa o fim de uma linha de texto e o início de outra.
- Um controle de retorno de linha, que move ao início de uma linha de texto (geralmente após um caractere de avanço de linha).
- Um controle de escape, que é adicionado automaticamente aos caracteres indicados na lista anterior.

## Exemplos de mensagens

Esta seção contém vários exemplos de mensagens SMS. Para cada exemplo, a seção mostra o número total de caracteres, bem como o número de partes da mensagem.

Exemplo 1: uma mensagem longa que contém somente caracteres no alfabeto GSM 03.38

A mensagem a seguir contém somente caracteres que estão no alfabeto GSM 03.38.

```
Hello Carlos. Your Example Corp. bill of $100 is now available. Autopay is
scheduled for next Thursday, April 9. To view the details of your bill, go
to https://example.com/bill1.
```

A mensagem anterior contém 180 caracteres, portanto, deve ser dividida em várias partes. Quando uma mensagem é dividida em várias partes, cada parte pode conter 153 caracteres GSM 03.38. Como resultado, esta mensagem será enviada em 2 partes.

Exemplo 2: uma mensagem que contém caracteres multibyte

A mensagem a seguir contém vários caracteres chineses, todos fora do alfabeto GSM 03.38.

```
#####.#####1994#7#####
```

A mensagem anterior contém 71 caracteres. No entanto, como quase todos os caracteres na mensagem estão fora do alfabeto GSM 03.38, ela será enviada em duas partes. Cada uma dessas partes da mensagem pode conter um máximo de 67 caracteres.

### Exemplo 3: uma mensagem que contém um único caractere que não faz parte do GSM

A mensagem a seguir contém um único caractere que não faz parte do alfabeto GSM 03.38. Neste exemplo, o caractere é uma aspa simples de fechamento ('), que é um caractere diferente de um apóstrofo regular ('). Aplicativos de processamento de texto, como o Microsoft Word, muitas vezes substituem automaticamente os apóstrofos por aspas simples de fechamento. Se fizer rascunhos de suas mensagens SMS no Microsoft Word e colá-las no Amazon SNS, você deverá remover esses caracteres especiais e substituí-los por apóstrofos.

```
John: Your appointment with Dr. Salazar's office is scheduled for next
Thursday at 4:30pm. Reply YES to confirm, NO to reschedule.
```

A mensagem anterior contém 130 caracteres. No entanto, como ela contém o caractere de aspas simples de fechamento, que não faz parte do alfabeto GSM 03.38, ela será enviada em duas partes.

Se você substituir o caractere de aspas simples de fechamento nesta mensagem por um apóstrofo (que faz parte do alfabeto GSM 03.38), a mensagem será enviada em uma única parte.

## Notificações por push para dispositivos móveis

Com o [Amazon SNS](#), você pode enviar mensagens de notificação por push diretamente para aplicações em dispositivos móveis. As mensagens de notificação por push enviadas para um endpoint móvel podem aparecer no aplicativo móvel como alertas de mensagem, atualizações de distintivo ou até mesmo como alertas com som.

### Tópicos

- [Como funcionam as notificações ao usuário](#)
- [Visão geral do processo de notificação ao usuário](#)
- [Configurar uma aplicação móvel](#)
- [Enviar notificações por push para dispositivos móveis](#)
- [Atributos de aplicativo móvel](#)
- [Eventos de aplicativo móvel](#)
- [Ações da API de push para dispositivos móveis](#)
- [Erros de API de push para dispositivos móveis](#)
- [Usar atributo de mensagem Time to Live \(TTL\) do Amazon SNS para notificações por push para dispositivos móveis](#)

- [Regiões compatíveis com aplicações móveis](#)
- [Práticas recomendadas para notificações por push móveis](#)

## Como funcionam as notificações ao usuário

Você envia as mensagens de notificação por push para dispositivos móveis e desktops usando um dos seguintes serviços de notificação por push compatíveis:

- Amazon Device Messaging (ADM)
- Apple Push Notification Service (APNs) para iOS e Mac OS X
- Baidu Cloud Push (Baidu)
- Firebase Cloud Messaging (FCM)
- Microsoft Push Notification Service for Windows Phone (MPNS)
- Windows Push Notification Services (WNS)

Os serviços de notificação por push, como o APNs e o FCM, mantêm uma conexão com cada aplicação e o respectivo dispositivo móvel registrado para usar o serviço. Quando uma aplicação e um dispositivo móvel são registrados, o serviço de notificação por push retorna um token de dispositivo. O Amazon SNS usa o token de dispositivo para criar um endpoint móvel, para o qual é possível enviar mensagens diretas de notificação por push. Para que o Amazon SNS se comunique com os diferentes serviços de notificação por push, você envia suas credenciais de serviço de notificação por push ao Amazon SNS para que sejam usadas em seu nome. Para ter mais informações, consulte [Visão geral do processo de notificação ao usuário](#).

Além de enviar mensagens de notificação por push diretas, você também pode usar o Amazon SNS para enviar mensagens para endpoints móveis inscritos em um tópico. O conceito é o mesmo que para inscrever outros tipos de endpoint, como Amazon SQS, HTTP/S, e-mail e SMS, em um tópico, conforme descrito em [O que é o Amazon SNS?](#). A diferença é que o Amazon SNS se comunica usando os serviços de notificação por push para que os endpoints móveis inscritos recebam mensagens de notificação por push enviadas para o tópico.

## Visão geral do processo de notificação ao usuário

1. [Obtenha as credenciais e o token de dispositivo](#) para as plataformas móveis para as quais deseja oferecer suporte.

2. Use as credenciais para criar um objeto de aplicativo da plataforma (`PlatformApplicationArn`) usando o Amazon SNS. Para ter mais informações, consulte [Criar uma aplicação de plataforma](#).
3. Use as credenciais retornadas para solicitar um token para o dispositivo e aplicativo móvel dos serviços de notificação por push. O token que você recebe representa seu dispositivo e aplicativo móvel.
4. Use o token de dispositivo e o `PlatformApplicationArn` para criar um objeto de endpoint da plataforma (`EndpointArn`) usando o Amazon SNS. Para ter mais informações, consulte [Criar um endpoint de plataforma](#).
5. Use o `EndpointArn` para [publicar uma mensagem em um aplicativo em um dispositivo móvel](#). Para obter mais informações, consulte [Publicar em um dispositivo móvel](#) e a API [Publish](#) na Referência da API do Amazon Simple Notification Service.

## Configurar uma aplicação móvel

Esta seção descreve como usar o AWS Management Console com as informações descritas em [Pré-requisitos para notificações ao usuário do Amazon SNS](#) para configurar aplicativos móveis.

### Tópicos

- [Pré-requisitos para notificações ao usuário do Amazon SNS](#)
- [Criar uma aplicação de plataforma](#)
- [Criar um endpoint de plataforma](#)
- [Adicionar tokens de dispositivo ou IDs de registro](#)
- [Métodos de autenticação da Apple](#)
- [Métodos de autenticação do Firebase Cloud Messaging \(FCM\)](#)
- [Gerenciamento de endpoints do Firebase Cloud Messaging \(FCM\)](#)

## Pré-requisitos para notificações ao usuário do Amazon SNS

Para começar a usar as notificações por push para dispositivos móveis do Amazon SNS, é necessário o seguinte:

- Um conjunto de credenciais para se conectar a um dos serviços de notificação por push compatíveis: ADM, APNs, Baidu, FCM, MPNS ou WNS.

- Um token de dispositivo ou ID de registro para o dispositivo e aplicativo móvel.
- Amazon SNS configurado para enviar mensagens de notificação por push para os endpoints móveis.
- Um aplicativo móvel que está registrado e configurado para usar um dos serviços de notificações por push compatíveis.

O registro da aplicação em um serviço de notificação por push requer várias etapas. O Amazon SNS precisa de algumas das informações que você fornece ao serviço de notificação por push para enviar mensagens de notificação por push diretas ao endpoint móvel. De modo geral, você precisa das credenciais necessárias para a conexão com o serviço de notificação por push, um token de dispositivo ou um ID de registro (que representam o dispositivo e o aplicativo móvel) recebido do serviço de notificação por push, e o aplicativo móvel registrado no serviço de notificação por push.

O formato exato das credenciais difere entre plataformas móveis, mas, em cada caso, essas credenciais devem ser enviadas durante uma conexão com a plataforma. Um conjunto de credenciais é emitido para cada aplicativo móvel e deve ser usado para enviar uma mensagem para qualquer instância do aplicativo.

Os nomes específicos variarão de acordo com qual serviço de notificação por push estiver sendo usado. Por exemplo, ao usar o APNs como o serviço de notificação por push, você precisa de um token de dispositivo. Como alternativa, ao usar o FCM, o token de dispositivo equivalente será chamado de ID de registro. O token de dispositivo ou o ID de registro é uma string enviada para o aplicativo pelo sistema operacional do dispositivo móvel. Ela identifica unicamente uma instância de um aplicativo móvel em execução em um determinado dispositivo móvel e pode ser considerada como identificadores exclusivos desse par de dispositivo e aplicativo.

O Amazon SNS armazena as credenciais (além de outras configurações) como um recurso de aplicação de plataforma. Os tokens de dispositivo (novamente com algumas configurações extras) são representados como objetos chamados endpoints de plataforma. Cada endpoint de plataforma pertence a um aplicativo de plataforma específico, e cada endpoint de plataforma pode ser acessado usando as credenciais armazenadas em seu aplicativo de plataforma correspondente.

As seções a seguir incluem os pré-requisitos para cada um dos serviços de notificação por push com suporte. Quando você tiver obtido as informações de pré-requisitos, poderá enviar uma mensagem de notificação por push usando o AWS Management Console ou as APIs de push para dispositivos móveis do Amazon SNS. Para mais informações, consulte [Visão geral do processo de notificação ao usuário](#).

## Criar uma aplicação de plataforma

Para que o Amazon SNS envie mensagens de notificação para endpoints móveis, seja diretamente ou por meio de assinaturas de um tópico, você precisa criar uma aplicação de plataforma primeiro. Após o registro da aplicação com AWS, a próxima etapa é criar um endpoint para a aplicação e o dispositivo móvel. O Amazon SNS usa então esse endpoint para enviar mensagens de notificação à aplicação e ao dispositivo.

Para criar uma aplicação de plataforma

1. Faça login no [console do Amazon SNS](#).
2. No painel de navegação, selecione Mobile (Dispositivo móvel) e escolha Push notifications (Notificações por push).
3. Na seção Platform applications (Aplicações de plataforma), selecione Create platform application (Criar aplicação de plataforma).

Para obter uma lista das regiões da AWS em que você pode criar aplicações para dispositivos móveis, consulte [Regiões compatíveis com aplicações móveis](#).

4. Em Application name (Nome da aplicação), insira um nome para representar sua aplicação.

Os nomes das aplicações devem ser compostos somente por letras maiúsculas e minúsculas, números, sublinhados, hifens e pontos. Os nomes também devem conter de 1 a 256 caracteres.

5. Em Push notification platform (Plataforma de notificações por push), selecione aquela na qual a aplicação está registrada e insira as credenciais apropriadas.

### Note

Se estiver usando uma das plataformas do Apple Push Notification Service (APNs), você poderá escolher entre [autenticação baseada em token ou certificado](#), selecionar Choose file (Escolher arquivo) para carregar o arquivo .p8 ou .p12 (exportado do Keychain Access) para o Amazon SNS.

6. Escolha Create platform application (Criar aplicativo de plataforma).

Isso registra a aplicação junto ao Amazon SNS, o que cria um objeto de aplicativo de plataforma para a plataforma selecionada e retorna um PlatformApplicationArn correspondente.

## Criar um endpoint de plataforma

Quando uma aplicação e um dispositivo móvel são registrados junto a um serviço de notificação por push, esse serviço retorna um token de dispositivo. O Amazon SNS usa o token de dispositivo para criar um endpoint móvel, para o qual é possível enviar mensagens diretas de notificação por push. Para ter mais informações, consulte [Pré-requisitos para notificações ao usuário do Amazon SNS](#) e [Visão geral do processo de notificação ao usuário](#).

Esta seção descreve a abordagem recomendada para a criação de um endpoint de plataforma.

### Tópicos

- [Crie um endpoint de plataforma](#)
- [Pseudocódigo](#)
- [AWS Exemplo de SDK](#)
- [Solução de problemas](#)

### Crie um endpoint de plataforma

Para enviar notificações para uma aplicação com o Amazon SNS, o token do dispositivo da aplicação deve primeiro ser registrado no Amazon SNS chamando a ação de criação de endpoint de plataforma. Essa ação leva o Nome de recurso da Amazon (ARN) do aplicativo de plataforma e o token do dispositivo como parâmetros e retorna o ARN do endpoint de plataforma criado.

A [CreatePlatformEndpoint](#) ação faz o seguinte:

- Se o endpoint de plataforma já existir, não o crie novamente. Retorne o ARN do endpoint de plataforma existente para o chamador.
- Se já existir um endpoint de plataforma com o mesmo token de dispositivo, mas com configurações diferentes, não crie o endpoint novamente. Gere uma exceção para o chamador.
- Se o endpoint de plataforma não existir, crie-o. Retorne o ARN do endpoint de plataforma recém-criado para o chamador.

Você não deve chamar a ação de criação do endpoint de plataforma imediatamente sempre que um aplicativo é iniciado, pois essa abordagem nem sempre fornece um endpoint que funciona. Isso pode acontecer, por exemplo, quando um aplicativo for desinstalado e reinstalado no mesmo dispositivo e o endpoint dele já existir mas estiver desabilitado. Para realizar um processo de registro bem-sucedido, você deve fazer o seguinte:

1. Verifique se um endpoint de plataforma existe para esta combinação de aplicativo-dispositivo.
2. Verifique se o token do dispositivo no endpoint de plataforma é o token do dispositivo válido mais recente.
3. Verifique se o endpoint de plataforma está ativado e pronto para uso.

## Pseudocódigo

O seguinte pseudocódigo descreve uma prática recomendada para a criação de um endpoint de plataforma atual, em funcionamento e habilitado em uma ampla variedade de condições de inicialização. Essa abordagem funciona se essa for a primeira vez que o aplicativo estiver sendo registrado ou não, se o endpoint de plataforma para este aplicativo já existir e se o endpoint de plataforma estiver ativado, tiver o token de dispositivo correto e assim por diante. É seguro chamá-lo várias vezes consecutivas, já que ele não criará endpoints de plataforma duplicados ou alterará um endpoint de plataforma existente se ele já estiver atualizado e ativado.

```
retrieve the latest device token from the mobile operating system
if (the platform endpoint ARN is not stored)
    # this is a first-time registration
    call create platform endpoint
    store the returned platform endpoint ARN
endif

call get endpoint attributes on the platform endpoint ARN

if (while getting the attributes a not-found exception is thrown)
    # the platform endpoint was deleted
    call create platform endpoint with the latest device token
    store the returned platform endpoint ARN
else
    if (the device token in the endpoint does not match the latest one) or
        (get endpoint attributes shows the endpoint as disabled)
        call set endpoint attributes to set the latest device token and then enable the
        platform endpoint
    endif
endif
```

Essa abordagem pode ser usada a qualquer momento que o aplicativo precisar se registrar ou efetuar um novo registro. Ela também pode ser usada ao notificar o Amazon SNS sobre uma alteração no token do dispositivo. Nesse caso, você pode chamar a ação com o valor de token do dispositivo mais recente. Alguns pontos a observar nessa abordagem são:

- Há dois casos em que ela pode chamar a ação de criação de endpoint de plataforma. Ela poderá ser chamada no início, quando o aplicativo não souber seu próprio ARN de endpoint de plataforma, como acontece durante um registro feito pela primeira vez. Ela também será chamada se a chamada inicial de ação de obtenção de atributos de endpoint falhar com uma exceção não encontrada. Isso também acontecerá se o aplicativo souber o ARN de endpoint, mas ele foi excluído.
- A ação de obtenção de atributos de endpoint é chamada para verificar o estado do endpoint de plataforma, mesmo se o endpoint de plataforma for recém-criado. Isso acontece quando o endpoint de plataforma já existe mas está desativado. Nesse caso, a ação de criação de endpoint de plataforma é bem-sucedida, mas não habilita o endpoint de plataforma. Portanto, você deve verificar o estado do endpoint da plataforma antes de retornar o sucesso.

## AWS Exemplo de SDK

O código a seguir mostra como implementar o pseudocódigo anterior usando os clientes do Amazon SNS fornecidos pelos AWS SDKs.

Para usar um AWS SDK, você deve configurá-lo com suas credenciais. Para obter mais informações, consulte [The shared config and credentials files](#) (“Os arquivos compartilhados de configuração e de credenciais”) no Guia de referência de AWS SDKs e ferramentas.

## CLI

### AWS CLI

Para criar um endpoint de aplicação de plataforma

O exemplo `create-platform-endpoint` a seguir cria um endpoint para a aplicação de plataforma especificada usando o token especificado.

```
aws sns create-platform-endpoint \  
  --platform-application-arn arn:aws:sns:us-west-2:123456789012:app/GCM/  
MyApplication \  
  --token EXAMPLE12345...
```

Saída:

```
{  
  "EndpointArn": "arn:aws:sns:us-west-2:1234567890:endpoint/GCM/  
MyApplication/12345678-abcd-9012-efgh-345678901234"
```

```
}
```

## Java

### SDK para Java 2.x

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.sns.SnsClient;
import software.amazon.awssdk.services.sns.model.CreatePlatformEndpointRequest;
import software.amazon.awssdk.services.sns.model.CreatePlatformEndpointResponse;
import software.amazon.awssdk.services.sns.model.SnsException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
 * started.html
 *
 * In addition, create a platform application using the AWS Management Console.
 * See this doc topic:
 *
 * https://docs.aws.amazon.com/sns/latest/dg/mobile-push-send-register.html
 *
 * Without the values created by following the previous link, this code examples
 * does not work.
 */

public class RegistrationExample {
    public static void main(String[] args) {
        final String usage = ""

                Usage:      <token> <platformApplicationArn>
```

```
        Where:
            token - The name of the FIFO topic.\s
            platformApplicationArn - The ARN value of platform
application. You can get this value from the AWS Management Console.\s
        """;

    if (args.length != 2) {
        System.out.println(usage);
        System.exit(1);
    }

    String token = args[0];
    String platformApplicationArn = args[1];
    SnsClient snsClient = SnsClient.builder()
        .region(Region.US_EAST_1)
        .build();

    createEndpoint(snsClient, token, platformApplicationArn);
}

public static void createEndpoint(SnsClient snsClient, String token, String
platformApplicationArn) {
    System.out.println("Creating platform endpoint with token " + token);
    try {
        CreatePlatformEndpointRequest endpointRequest =
CreatePlatformEndpointRequest.builder()
            .token(token)
            .platformApplicationArn(platformApplicationArn)
            .build();

        CreatePlatformEndpointResponse response =
snsClient.createPlatformEndpoint(endpointRequest);
        System.out.println("The ARN of the endpoint is " +
response.endpointArn());
    } catch (SnsException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}
}
```

Para ter mais informações, consulte [Ações da API de push para dispositivos móveis](#).

## Solução de problemas

Chamar repetidamente a criação de endpoint da plataforma com um token de dispositivo desatualizado

Especialmente para endpoints do FCM, você pode achar melhor armazenar o primeiro token de dispositivo emitido pelo aplicativo e, em seguida, chamar o endpoint de criação da plataforma com esse token de dispositivo toda vez que o aplicativo for inicializado. Isso pode parecer correto, pois a aplicação não precisará gerenciar o estado do token do dispositivo, e o Amazon SNS atualizará automaticamente o token do dispositivo para seu valor mais recente. No entanto, essa solução tem uma série de problemas sérios:

- O Amazon SNS utiliza o feedback do FCM para atualizar os tokens de dispositivo expirados para novos tokens de dispositivo. O FCM retém informações sobre tokens de dispositivo antigos por algum tempo, mas não indefinidamente. Quando o FCM esquecer a conexão entre o token do dispositivo antigo e o novo token, o Amazon SNS não poderá mais atualizar o token do dispositivo armazenado no endpoint de plataforma para seu valor correto, ele apenas desabilitará o endpoint de plataforma.
- O aplicativo de plataforma conterá vários endpoints de plataforma correspondentes ao mesmo token de dispositivo.
- O Amazon SNS impõe uma cota para o número de endpoints de plataforma que podem ser criados a partir do mesmo token de dispositivo. Finalmente, a criação de novos endpoints falhará com uma exceção de parâmetro inválido e aparecerá a seguinte mensagem de erro: “This endpoint is already registered with a different token.”

Para obter mais informações sobre o gerenciamento de endpoints do FCM, consulte [Gerenciamento de endpoints do Firebase Cloud Messaging \(FCM\)](#)

Habilitar novamente um endpoint de plataforma associado a um token de dispositivo inválido

Quando uma plataforma móvel (como o APNs ou o FCM) informa ao Amazon SNS que o token do dispositivo usado na solicitação de publicação é inválido, o Amazon SNS desativa o endpoint de plataforma associado a esse token de dispositivo. O Amazon SNS rejeitará as publicações posteriores nesse token de dispositivo. Você pode achar melhor simplesmente reativar o endpoint de plataforma e manter a publicação, mas, na maioria dos casos, isso não funcionará: as mensagens

que são publicadas não são entregues e o endpoint de plataforma é desativado novamente logo em seguida.

Isso ocorre porque o token de dispositivo associado ao endpoint de plataforma é genuinamente inválido. As entregas destinadas a ele não terão sucesso porque ele não corresponde a nenhum aplicativo instalado. Na próxima vez em que for publicada, a plataforma móvel informará novamente ao Amazon SNS que o token de dispositivo é inválido, e o Amazon SNS desabilitará novamente o endpoint de plataforma.

Para ativar novamente um endpoint de plataforma desabilitada, ele precisa ser associado a um token de dispositivo válido (com uma chamada de ação de definição de atributos de endpoint) e depois habilitado. A partir de então, as entregas para o endpoint dessa plataforma terão sucesso. A única vez que a reabilitação de um endpoint de plataforma sem atualização do seu token de dispositivo funcionará será quando um token de dispositivo associado a esse endpoint for inválido, mas depois tornar-se válido novamente. Isso pode acontecer, por exemplo, quando um aplicativo for desinstalado e instalado novamente no mesmo dispositivo móvel e receber o mesmo token de dispositivo. A abordagem apresentada anteriormente faz isso, garantindo que a reativação de um endpoint de plataforma seja feita somente após verificar se o token de dispositivo associado a ele é o mais atual disponível.

## Adicionar tokens de dispositivo ou IDs de registro

Quando você registra um aplicativo e dispositivo móvel com um serviço de notificação pela primeira vez, como o Apple Push Notification Service (APNs) e o Firebase Cloud Messaging (FCM), os tokens de dispositivo ou IDs de registro são retornados pelo serviço de notificação. Quando você adiciona os tokens de dispositivo ou os IDs de registro ao Amazon SNS, eles são usados com a API `PlatformApplicationArn` para criar um endpoint para a aplicação e o dispositivo. Quando o Amazon SNS cria o endpoint, um `EndpointArn` é retornado. O `EndpointArn` é como o Amazon SNS sabe a qual aplicação e dispositivo móvel enviar a mensagem de notificação.

Você pode adicionar tokens de dispositivo e IDs de registro ao Amazon SNS usando os seguintes métodos:

- Adicionar manualmente um único token à AWS usando o AWS Management Console
- Fazer upload de vários tokens usando a API `CreatePlatformEndpoint`
- Registrar tokens de dispositivos que instalarão seus aplicativos no futuro

Para adicionar manualmente um token de dispositivo ou ID de registro

1. Faça login no [console do Amazon SNS](#).
2. Selecione Dispositivo móvel e escolha Notificações por push.
3. Na seção Aplicações de plataforma, selecione sua aplicação e escolha Editar. Se você ainda não criou uma aplicação de plataforma, crie uma agora. Para obter instruções sobre como fazer isso, consulte [Criar uma aplicação de plataforma](#).
4. Escolha Adicionar endpoints.
5. Na caixa Token do endpoint, digite o ID do token ou o ID de registro, dependendo do serviço de notificação usado. Por exemplo, para o ADM e o FCM, informe o ID de registro.
6. (Opcional) Na caixa User Data (Dados do usuário), digite informações aleatórias a serem associadas ao endpoint. O Amazon SNS não usa esses dados. Os dados devem ser no formato UTF-8 e ter menos de 2 KB.
7. Finalmente, escolha Add Endpoints (Adicionar endpoints).

Agora, com o endpoint criado, você pode enviar mensagens diretamente para um dispositivo móvel ou enviar mensagens para dispositivos móveis que estejam inscritos em um tópico.

Para fazer o upload de vários tokens usando a API **CreatePlatformEndpoint**

As etapas a seguir mostram como usar a aplicação Java de exemplo (pacote `bulkupload`) fornecida pela AWS para fazer upload de vários tokens (tokens de dispositivo ou IDs de registro) no Amazon SNS. Você pode usar esse aplicativo de amostra para ajudá-lo a começar a fazer upload de tokens existentes.

#### Note

As etapas a seguir usam o IDE para Java Eclipse. As etapas presumem que você já tenha instalado o AWS SDK for Java e que também tenha as credenciais de segurança da AWS para sua Conta da AWS. Para obter mais informações, consulte [AWS SDK for Java](#). Para obter mais informações sobre credenciais, consulte [Como obter credenciais de segurança?](#) na Referência geral da AWS.

1. Faça o download e descompacte o arquivo [snsmobilepush.zip](#).
2. Crie um novo projeto Java no Eclipse.

3. Importe a pasta `SNSSamples` para o diretório de nível superior do projeto Java recém-criado. No Eclipse, escolha com o botão direito do mouse o nome do projeto Java, escolha `Import` (Importar), expanda `General` (Geral), escolha `File System` (Sistema de arquivos), escolha `Next` (Avançar), navegue até a pasta `SNSSamples`, escolha `OK` e escolha `Finish` (Concluir).
4. Faça o download de uma cópia da biblioteca [OpenCSV](#) e adicione-a ao `Build Path` do pacote `bulkupload`.
5. Abra o arquivo `BulkUpload.properties` contido no pacote `bulkupload`.
6. Adicione o seguinte ao `BulkUpload.properties`:
  - O `ApplicationArn` para o qual você deseja adicionar endpoints.
  - O caminho absoluto para o local do arquivo CSV que contém os tokens.
  - Os nomes dos arquivos CSV (como `goodTokens.csv` e `badTokens.csv`) que deverão ser criados para registrar em log os tokens que o Amazon SNS analisa corretamente e aqueles que falham.
  - (Opcional) Os caracteres para especificar o delimitador e a cota no arquivo CSV que contém os tokens.
  - (Opcional) O número de threads para usar a fim de criar simultaneamente endpoints. O padrão é 1 thread.

O `BulkUpload.properties` completo deve ser semelhante ao seguinte:

```
applicationarn:arn:aws:sns:us-west-2:111122223333:app/FCM/fcmpushapp
csvfilename:C:\\mytokendirectory\\mytokens.csv
goodfilename:C:\\mylogfiles\\goodtokens.csv
badfilename:C:\\mylogfiles\\badtokens.csv
delimiterchar:'
quotechar:"
numofthreads:5
```

7. Execute a aplicação `BatchCreatePlatformEndpointSample.java` para fazer o upload dos tokens para o Amazon SNS.

Neste exemplo, os endpoints que foram criados para os tokens carregados com êxito no Amazon SNS serão registrados em log no `goodTokens.csv`, enquanto os tokens com defeito serão registrados em log no `badTokens.csv`. Além disso, você deve ver logs `STD OUT` gravados no console do Eclipse, com conteúdo semelhante ao seguinte:

```
<1>[SUCCESS] The endpoint was created with Arn arn:aws:sns:us-west-2:111122223333:app/FCM/fcmpushapp/165j2214-051z-3176-b586-138o3d420071
<2>[ERROR: MALFORMED CSV FILE] Null token found in /mytokendirectory/mytokens.csv
```

Para registrar tokens de dispositivos que instalarão seus aplicativos no futuro

Você pode usar uma das duas opções a seguir:

- Use the Amazon Cognito service (Usar o serviço Amazon Cognito): seu aplicativo móvel precisará de credenciais para criar endpoints associados à aplicação de plataforma do Amazon SNS. Recomendamos o uso de credenciais temporárias que expirem após um período de tempo. Para a maioria dos cenários, recomendamos que você use o Amazon Cognito para criar credenciais de segurança temporárias. Para obter mais informações, consulte o [Guia do desenvolvedor do Amazon Cognito](#). Se desejar ser notificado quando um aplicativo for registrado com o Amazon SNS, você poderá se registrar para receber um evento do Amazon SNS que fornecerá o novo ARN do endpoint. Você também pode usar a API `ListEndpointByPlatformApplication` para obter a lista completa de endpoints registrados no Amazon SNS.
- Uso de um servidor de proxy: se a infraestrutura do aplicativo já estiver configurada para os aplicativos móveis notificarem e registrarem-se a cada instalação, você poderá continuar a usar essa configuração. Seu servidor age como um proxy e passa o token do dispositivo para notificações por push para dispositivos móveis do Amazon SNS, juntamente com os dados do usuário que você gostaria de armazenar. Para esse fim, o servidor de proxy se conectará ao Amazon SNS usando suas credenciais da AWS e usará a chamada de API `CreatePlatformEndpoint` para carregar as informações do token. O Amazon Resource Name (ARN – Nome de recurso Amazon) do endpoint recém-criado será retornado e poderá ser armazenado por seu servidor para fazer chamadas subsequentes para publicar no Amazon SNS.

## Métodos de autenticação da Apple

É possível autorizar o Amazon SNS a enviar notificações por push para sua aplicação iOS ou macOS fornecendo informações que identifiquem você como desenvolvedor da aplicação. Para autenticar, forneça uma chave ou um certificado [ao criar uma aplicação de plataforma](#). Ambos estão disponíveis em sua conta de Desenvolvedor da Apple.

## Chave de assinatura de token

Uma chave de assinatura privada que o Amazon SNS usa para assinar tokens de autenticação do Apple Push Notification Service (APNs).

Se você fornecer uma chave de assinatura, o Amazon SNS usará um token para autenticar com APNs para cada notificação por push enviada. Com a chave de assinatura, envie notificações por push a ambientes de produção de APNs e de sandbox.

Sua chave de assinatura não expira e a mesma chave de assinatura pode ser usada para várias aplicações. Para obter mais informações, consulte [Comunicação com o APNs usando tokens de autenticação](#) na seção Ajuda da conta de desenvolvedor no site da Apple.

## Certificado

Um certificado TLS que o Amazon SNS usa para autenticar com o APNs quando você envia notificações por push. O certificado pode ser obtido da sua conta de Desenvolvedor da Apple.

Os certificados expiram após um ano. Quando isso acontecer, crie um novo certificado e forneça-o ao Amazon SNS. Para obter mais informações, consulte [Establishing a Certificate-Based Connection to APNs](#) (Estabelecer uma conexão baseada em certificado com o APNs) no site Apple Developer.

Para gerenciar as configurações do APNs usando o Console de Gerenciamento da AWS

1. Faça login no [console do Amazon SNS](#).
2. Em Mobile (Dispositivos móveis), escolha Push notifications (Notificações por push).
3. Selecione a Application (Aplicação) que gostaria de editar nas configurações do APNs, em seguida escolha Edit (Editar).
4. Na página Edit (Editar), em Authentication type (Tipo de autenticação), escolha Token ou Certificate (Certificado).
5. Carregue as credenciais apropriadas para a chave de assinatura do certificado ou do token. Você pode obter essas informações em sua conta de desenvolvedor da Apple.
6. Dependendo do tipo de autenticação escolhido, siga um destes procedimentos:
  - Se escolher Token, forneça as seguintes informações da sua conta de Desenvolvedor da Apple. O Amazon SNS exige essas informações para construir tokens de autenticação.

- **Signing key (Chave de assinatura):** a chave de assinatura do token de autenticação da conta de Desenvolvedor da Apple, que você baixa como um arquivo.p8. A Apple permite que você baixe a chave de assinatura apenas uma vez.
- **Signing key ID (ID da chave de assinatura):** o ID atribuído a sua chave de assinatura. O Amazon SNS exige essas informações para construir tokens de autenticação. Para localizar esse valor em sua conta de Desenvolvedor da Apple, escolha Certificates, IDs & Profiles (Certificados, IDs e perfis) e escolha a chave na seção Keys (Chaves).
- **Team identifier (Identificador da equipe):** o ID atribuído à sua equipe de conta de Desenvolvedor da Apple. Você pode encontrar esse valor na página Membership (Associação).
- **Bundle identifier (Identificador do pacote):** o ID atribuído à sua aplicação. Para localizar esse valor, selecione Certificates, IDs & Profiles (Certificados, IDs e perfis), selecione App IDs (IDs da aplicação) na seção Identifiers (Identificadores) e selecione a aplicação.
- Se você escolher Certificate (Certificado), forneça as seguintes informações:
  - **SSL certificate (Certificado SSL):** o arquivo .p12 do certificado TLS. Exporte esse arquivo de Keychain Access após baixar e instalar o certificado de sua conta de Desenvolvedor da Apple.
  - **Certificate password (Senha do certificado):** se você atribuiu uma senha ao certificado, especifique-a aqui.

7. Quando terminar, escolha Salvar alterações.

## Métodos de autenticação do Firebase Cloud Messaging (FCM)

Este tópico descreve como obter as credenciais necessárias da API do FCM (HTTP v1) do Google para usar com a AWS API e a AWS CLI AWS Management Console

### Tópicos

- [Pré-requisito](#)
- [Gerenciamento das configurações do FCM \(API\)](#)
- [Gerenciar configurações do FCM \(CLI\)](#)
- [Gerenciar configurações do FCM \(console\)](#)

### Important

20 de junho de 2023 — O Google suspendeu o uso da API HTTP legada do Firebase Cloud Messaging (FCM). O Amazon SNS agora oferece suporte à entrega para todos os tipos de dispositivos usando a API HTTP v1 do FCM. Recomendamos que você migre seus aplicativos push móveis existentes para a API HTTP v1 mais recente do FCM até 1º de junho de 2024 para evitar interrupções.

18 de janeiro de 2024 — O Amazon SNS introduziu o suporte à API HTTP v1 do FCM para entrega de notificações push móveis para dispositivos Android.

26 de março de 2024 — O Amazon SNS oferece suporte à API HTTP v1 do FCM para dispositivos Apple e destinos Webpush. Recomendamos que você migre seus aplicativos push móveis existentes para a API HTTP v1 mais recente do FCM até 1º de junho de 2024 para evitar interrupções nos aplicativos.

É possível autorizar o Amazon SNS a enviar notificações por push para suas aplicações fornecendo informações que identifiquem você como desenvolvedor da aplicação. Para autenticar, forneça uma chave de API ou um token [ao criar uma aplicação de plataforma](#). Você pode obter as seguintes informações no [console de aplicativos do Firebase](#):

#### Chave de API

A chave de API é uma credencial usada ao chamar a API legada do Firebase. As APIs legadas do FCM serão removidas pelo Google em 20 de junho de 2024. Se você estiver usando uma chave de API como credencial da plataforma, atualize a credencial da plataforma selecionando Token como opção e fazendo upload do arquivo JSON associado à aplicação Firebase.

#### Token

Um token de acesso de curta duração é usado ao chamar a API HTTP v1. Essa é a API sugerida pelo Firebase para enviar notificações por push. Para gerar tokens de acesso, o Firebase fornece aos desenvolvedores um conjunto de credenciais na forma de um arquivo de chave privada (também conhecido como arquivo `service.json`).

#### Pré-requisito

Você deve obter as credenciais `service.json` do FCM antes de começar a gerenciar as configurações do FCM no Amazon SNS. Para obter suas credenciais `service.json`, consulte [Migrar de HTTP legado para HTTP v1](#) na documentação do Firebase do Google.

## Gerenciamento das configurações do FCM (API)

Você pode criar notificações push do FCM usando a AWS API. O número e o tamanho dos recursos do Amazon SNS em uma AWS conta são limitados. Para obter mais informações, consulte os [endpoints e cotas do Amazon Simple Notification Service](#) no Referência geral da AWS Guia.

Para criar uma notificação push do FCM junto com um tópico AWS (API) do Amazon SNS

Ao usar as credenciais de chave, `PlatformCredential` é API key. Ao usar credenciais de token, `PlatformCredential` é um arquivo de chave privada formatado em JSON:

- [CreatePlatformApplication](#)

Para recuperar um tipo de credencial do FCM para um tópico existente do Amazon SNS (API)AWS

Recupera o tipo de credencial "AuthenticationMethod": "Token" ou "AuthenticationMethod": "Key":

- [GetPlatformApplicationAttributes](#)

Como definir um atributo do FCM para um tópico existente do Amazon SNS (AWS API)

Define o atributo do FCM:

- [SetPlatformApplicationAttributes](#)

## Gerenciar configurações do FCM (CLI)

Você pode criar notificações push do FCM usando a AWS Command Line Interface (CLI). O número e o tamanho dos recursos do Amazon SNS em uma AWS conta são limitados. Para obter mais informações, consulte [Amazon Simple Notification Service endpoints and quotas](#) (Endpoints e cotas do Amazon Simple Notification Service).

Como criar uma notificação por push do FCM junto com um tópico do Amazon SNS (AWS CLI)

Ao usar as credenciais de chave, `PlatformCredential` é API key. Ao usar credenciais de token, `PlatformCredential` é um arquivo de chave privada formatado em JSON: Ao usar a AWS CLI, o arquivo deve estar no formato de string e os caracteres especiais devem ser ignorados. Para formatar o arquivo corretamente, o Amazon SNS recomenda usar o seguinte comando:  
`SERVICE_JSON=`jq @json <<< cat service.json``:

- [create-platform-application](#)

Como recuperar um tipo de credencial do FCM para um tópico existente do Amazon SNS (AWS CLI)

Recupera o tipo de credencial "AuthenticationMethod": "Token" ou "AuthenticationMethod": "Key":

- [get-platform-application-attributes](#)

Como definir um atributo do FCM para um tópico existente do Amazon SNS (AWS CLI)

Define o atributo do FCM:

- [set-platform-application-attributes](#)

Gerenciar configurações do FCM (console)

Use as etapas a seguir para inserir as credenciais que sua aplicação usa para se conectar ao FCM.

1. Faça login no console [do Amazon SNS](#).
2. Em Mobile (Dispositivos móveis), escolha Push notifications (Notificações por push).
3. Escolha uma aplicação do FCM existente e selecione Editar. Se você ainda não criou uma aplicação de plataforma, consulte [Criar uma aplicação de plataforma](#).
4. Na página Editar, para Credenciais do Firebase Cloud Messaging, escolha Token ou Chave. É possível consultar as informações a seguir no [console da aplicação do Firebase](#).
  - Se você escolher Token, faça upload de um arquivo de chave privada válido. O conteúdo desse arquivo é usado para gerar tokens de acesso de curta duração ao enviar notificações.
  - Se você escolher Chave, insira a chave da API do Google.
5. Quando terminar, escolha Salvar alterações.

Tópicos relacionados

- [Usando cargas do Google Firebase Cloud Messaging \(FCM\) v1 no Amazon SNS](#)

## Gerenciamento de endpoints do Firebase Cloud Messaging (FCM)

### Tópicos

- [Gerenciamento e manutenção de tokens de dispositivos](#)
- [Detectando tokens inválidos](#)
- [Removendo tokens obsoletos](#)

### Gerenciamento e manutenção de tokens de dispositivos

Você pode garantir a capacidade de entrega das notificações push do seu aplicativo móvel seguindo estas etapas:

1. Armazene todos os tokens do dispositivo, os ARNs de endpoint do Amazon SNS correspondentes e os carimbos de data/hora em seu servidor de aplicativos.
2. Remova todos os tokens obsoletos e exclua os ARNs correspondentes do endpoint do Amazon SNS.

Na inicialização inicial do seu aplicativo, você receberá um token de dispositivo (também conhecido como token de registro) para o dispositivo. Esse token de dispositivo é gerado pelo sistema operacional do dispositivo e está vinculado ao seu aplicativo FCM. Depois de receber esse token de dispositivo, você pode registrá-lo no Amazon SNS como um endpoint da plataforma. Recomendamos que você armazene o token do dispositivo, o ARN do endpoint da plataforma Amazon SNS e o carimbo de data/hora salvando-os em seu servidor de aplicativos ou em outro armazenamento persistente. Para configurar seu aplicativo FCM para recuperar e armazenar tokens de dispositivos, consulte [Recuperar e armazenar tokens de registro na documentação do Firebase do Google](#).

É importante que você mantenha up-to-date os tokens. Os tokens do dispositivo do seu usuário podem mudar sob as seguintes condições:

1. O aplicativo móvel é restaurado em um novo dispositivo.
2. O usuário desinstala ou atualiza o aplicativo.
3. O usuário limpa os dados do aplicativo.

Quando o token do seu dispositivo for alterado, recomendamos que você atualize o endpoint correspondente do Amazon SNS com o novo token. Isso permite que o Amazon SNS continue a comunicação com o dispositivo registrado. Você pode fazer isso implementando o seguinte

pseudocódigo em seu aplicativo móvel. Ele descreve uma prática recomendada para criar e manter endpoints de plataforma habilitados. Essa abordagem pode ser executada sempre que os aplicativos móveis são iniciados ou como um trabalho agendado em segundo plano.

## Pseudocódigo

Use o seguinte pseudocódigo do FCM para gerenciar e manter os tokens do dispositivo.

```
retrieve the latest token from the mobile OS
if (endpoint arn not stored)
    # first time registration
    call CreatePlatformEndpoint
    store returned endpoint arn
endif

call GetEndpointAttributes on the endpoint arn

if (getting attributes encountered NotFound exception)
    #endpoint was deleted
    call CreatePlatformEndpoint
    store returned endpoint arn
else
    if (token in endpoint does not match latest) or
        (GetEndpointAttributes shows endpoint as disabled)
        call SetEndpointAttributes to set the
            latest token and enable the endpoint
    endif
endif
endif
```

Para saber mais sobre os requisitos de atualização de tokens, consulte [Atualizar tokens regularmente](#) na documentação do Firebase do Google.

## Detectando tokens inválidos

Quando uma mensagem é enviada para um endpoint do FCM v1 com um token de dispositivo inválido, o Amazon SNS receberá uma das seguintes exceções:

- UNREGISTERED(HTTP 404) — Quando o Amazon SNS receber essa exceção, você receberá um evento de falha na entrega com *FailureType* um token *InvalidPlatformToken* de plataforma de e *FailureMessage* um de associado ao endpoint não é válido. O Amazon SNS desativará o endpoint da sua plataforma quando uma entrega falhar, com essa exceção.

- `INVALID_ARGUMENT(HTTP 400)` — Quando o Amazon SNS recebe essa exceção, significa que o token do dispositivo ou a carga útil da mensagem é inválida. Para obter mais informações, consulte [Error Code](#) documentação do Firebase do Google.

Como `INVALID_ARGUMENT` pode ser devolvido em qualquer um desses casos, o Amazon SNS retornará um `FailureType` de `InvalidNotification` e um corpo `FailureMessage` de notificação é inválido. Ao receber esse erro, verifique se sua carga está correta. Se estiver correto, verifique se o token do dispositivo está up-to-date. O Amazon SNS não desativará o endpoint da sua plataforma quando uma entrega falhar, com essa exceção.

Outro caso em que você enfrentará um evento de falha na `InvalidPlatformToken` entrega é quando o token do dispositivo registrado não pertence ao aplicativo que está tentando enviar essa mensagem. Nesse caso, o Google retornará um erro `SENDER_ID_MISMATCH`. O Amazon SNS desativará o endpoint da sua plataforma quando uma entrega falhar, com essa exceção.

Todos os códigos de erro observados recebidos da API FCM v1 estão disponíveis CloudWatch quando você configura o [registro do status de entrega do](#) seu aplicativo.

Para receber eventos de entrega para seu aplicativo, consulte [Eventos do aplicativo disponíveis](#).

### Removendo tokens obsoletos

Os tokens são considerados obsoletos quando as entregas de mensagens ao dispositivo terminal começam a falhar. O Amazon SNS define esses tokens obsoletos como endpoints desativados para seu aplicativo de plataforma. Quando você publica em um endpoint desativado, o Amazon SNS retornará `EventDeliveryFailure` um evento com `FailureType` o `EndpointDisabled` de, e `FailureMessage` um endpoint de é desativado. Para receber eventos de entrega para seu aplicativo, consulte [Eventos do aplicativo disponíveis](#).

Ao receber esse erro do Amazon SNS, você precisa remover ou atualizar o token obsoleto no aplicativo da sua plataforma.

## Enviar notificações por push para dispositivos móveis

Esta seção descreve como enviar notificações por push móveis.

### Tópicos

- [Publicar em um tópico](#)
- [Publicar em um dispositivo móvel](#)

- [Publicação com cargas úteis específicas da plataforma](#)

## Publicar em um tópico

Também é possível usar o Amazon SNS para enviar mensagens para endpoints móveis inscritos em um tópico. O conceito é o mesmo que para inscrever outros tipos de endpoint, como Amazon SQS, HTTP/S, e-mail e SMS, em um tópico, conforme descrito em [O que é o Amazon SNS?](#). A diferença é que o Amazon SNS se comunica por meio de serviços de notificação como o Apple Push Notification Service (APNS) e o Google Firebase Cloud Messaging (FCM). Por meio do serviço de notificações, os endpoints móveis inscritos recebem as notificações enviadas ao tópico.

## Publicar em um dispositivo móvel

Você pode enviar mensagens de notificação por push do Amazon SNS diretamente para um endpoint que representa uma aplicação em um dispositivo móvel.

Para enviar uma mensagem direta

1. Faça login no [console do Amazon SNS](#).
2. No painel de navegação, escolha Push notifications (Notificações por push).
3. Na página de notificações push móveis, na seção Aplicativos da plataforma, escolha o nome do aplicativo, por exemplo **MyApp**.
4. Na **MyApp** página, na seção Endpoints, escolha um endpoint e escolha Publicar mensagem.
5. Na página Publish message to endpoint (Publicar mensagem no endpoint), insira a mensagem que aparecerá no aplicativo do dispositivo móvel e escolha Publish message (Publicar mensagem).

O Amazon SNS envia a mensagem de notificação para o serviço de notificação da plataforma que, por sua vez, envia a mensagem para a aplicação.

## Publicação com cargas úteis específicas da plataforma

Você pode usar as AWS Management Console APIs do Amazon SNS para enviar mensagens personalizadas com cargas específicas da plataforma para dispositivos móveis. Para obter informações sobre a utilização das APIs do Amazon SNS, consulte [Ações da API de push para dispositivos móveis](#) e o arquivo SNSMobilePush.java em [snsmobilepush.zip](#).

## Tópicos

- [Enviar mensagens formatadas em JSON](#)
- [Enviar mensagens específicas à plataforma](#)
- [Enviar mensagens para um aplicativo em várias plataformas](#)
- [Enviar mensagens para o APNs como notificações de alerta ou de segundo plano](#)
- [Usando cargas do Google Firebase Cloud Messaging \(FCM\) v1 no Amazon SNS](#)

## Enviar mensagens formatadas em JSON

Ao enviar cargas específicas à plataforma, os dados devem estar formatados como strings de par de chave/valor JSON, com aspas em sequência de escape.

Os exemplos a seguir mostram uma mensagem personalizada para a plataforma do FCM.

```
{
  "GCM": "{\"fcmV1Message\": {\"message\": {\"notification\": {\"title\": \"Hello\",
    \"body\": \"This is a test.\"}, \"data\": {\"dataKey\": \"example\"}}}}"
```

## Enviar mensagens específicas à plataforma

Além de enviar dados personalizados como pares de chave/valor, é possível enviar pares de chave/valor específicos à plataforma.

O exemplo a seguir mostra a inclusão dos parâmetros `time_to_live` e `collapse_key` do FCM depois dos pares de chave/valor de dados personalizados no parâmetro `data` do FCM.

```
{
  "GCM": "{\"fcmV1Message\": {\"message\": {\"notification\": {\"title\": \"TitleTest\",
    \"body\": \"Sample message for Android or iOS endpoints.\"}, \"data\": {\"time_to_live\": 3600, \"collapse_key\": \"deals\"}}}}"
```

Para obter uma lista dos pares de chave/valor compatíveis em cada um dos serviços de notificação push compatíveis com o Amazon SNS, consulte o seguinte:

### Important

O Amazon SNS agora oferece suporte à API HTTP v1 do Firebase Cloud Messaging (FCM) para enviar notificações push móveis para dispositivos Android.

26 de março de 2024 — O Amazon SNS oferece suporte à API HTTP v1 do FCM para dispositivos Apple e destinos Webpush. Recomendamos que você migre seus aplicativos push móveis existentes para a API HTTP v1 mais recente do FCM até 1º de junho de 2024 para evitar interrupções nos aplicativos.

- [Payload Key Reference](#) (“Referência de chave da carga útil”) na documentação do APNs
- [Protocolo HTTP do Firebase Cloud Messaging](#) na documentação do FCM
- [Send a Message](#) (“Enviar uma mensagem”) na documentação do ADM

## Enviar mensagens para um aplicativo em várias plataformas

Para enviar uma mensagem para uma aplicação instalada em dispositivos para várias plataformas, como o FCM e o APNs, deverá primeiro ser necessário inscrever os endpoints móveis em um tópico do Amazon SNS e depois publicar a mensagem no tópico.

O exemplo seguinte mostra uma mensagem a ser enviada para endpoints móveis inscritos no APNs, no FCM e no ADM:

```
{
  "default": "This is the default message which must be present when publishing a
message to a topic. The default message will only be used if a message is not present
for
one of the notification platforms.",
  "APNS": "{\"aps\":{\"alert\": \"Check out these awesome deals!\",\"url\":
\"www.amazon.com\"} }",
  "GCM": "{\"data\":{\"message\": \"Check out these awesome deals!\",\"url\":
\"www.amazon.com\"}}",
  "ADM": "{\"data\":{\"message\": \"Check out these awesome deals!\",\"url\":
\"www.amazon.com\"}}"
}
```

## Enviar mensagens para o APNs como notificações de alerta ou de segundo plano

O Amazon SNS pode enviar mensagens para o APNs como notificações de alert ou de background (para obter mais informações, consulte [Pushing Background Updates to Your App](#) (“Enviar atualizações em segundo plano para sua aplicação”) na documentação do APNs.

- Uma notificação de alert do APNs informa o usuário exibindo uma mensagem de alerta, reproduzindo um som ou adicionando um distintivo ao ícone da aplicação.

- Uma notificação de background do APNs desperta ou instrui sua aplicação a agir em relação ao conteúdo da notificação, sem informar o usuário.

Especificar valores personalizados do cabeçalho do APNs

Recomendamos especificar valores personalizados para o [atributo de mensagem AWS.SNS.MOBILE.APNS.PUSH\\_TYPE reservada](#) usando a ação da API do Amazon Publish SNS AWS, SDKs ou o AWS CLI. O exemplo da CLI a seguir define `content-available` como 1 e `apns-push-type` como `background` para o tópico especificado.

```
aws sns publish \
--endpoint-url https://sns.us-east-1.amazonaws.com \
--target-arn arn:aws:sns:us-east-1:123456789012:endpoint/APNS_PLATFORM/MYAPP/1234a567-
bc89-012d-3e45-6fg7h890123i \
--message '{"APNS_PLATFORM":{"aps":{"content-available":1}}}' \
--message-attributes '{ \
  "AWS.SNS.MOBILE.APNS.TOPIC": \
{"DataType":"String","StringValue":"com.amazon.mobile.messaging.myapp"}, \
  "AWS.SNS.MOBILE.APNS.PUSH_TYPE":{"DataType":"String","StringValue":"background"} \
  "AWS.SNS.MOBILE.APNS.PRIORITY":{"DataType":"String","StringValue":"5"}}', \
--message-structure json
```

Inferir o cabeçalho do tipo de push do APNs da carga útil

Se você não definir o cabeçalho `apns-push-type` do APNs, o Amazon SNS definirá o cabeçalho como `alert` ou `background`, dependendo da chave `content-available` no dicionário `aps` da configuração de carga útil do APNs formatada em JSON.

#### Note

O Amazon SNS é capaz de inferir somente os cabeçalhos `alert` ou `background`, embora o cabeçalho `apns-push-type` possa ser definido com outros valores.

- `apns-push-type` é definido como `alert`
  - Se o dicionário `aps` contiver `content-available` definida como 1 e uma ou mais chaves que acionem interações do usuário.
  - Se o dicionário `aps` contiver `content-available` definida como 0 ou se a chave `content-available` estiver ausente.

- Se o valor da chave `content-available` não for um inteiro ou um booleano.
- `apns-push-type` é definido como `background`
- Se o dicionário `aps` contiver somente a variável `content-available` definida como `1` e nenhuma outra chave que acione interações com o usuário.

#### Important

Se o Amazon SNS enviar um objeto de configuração bruto para o APNs como uma notificação somente de segundo plano, será necessário incluir o `content-available` definido como `1` no dicionário `aps`. Embora você possa incluir chaves personalizadas, o dicionário `aps` não deve conter chaves que acionem interações do usuário (por exemplo, alertas, distintivos ou sons).

Veja a seguir um exemplo de objeto de configuração bruto.

```
{
  "APNS": "{\"aps\":{\"content-available\":1},\"Foo1\":\"Bar\",\"Foo2\":123}"
}
```

Neste exemplo, o Amazon SNS define o cabeçalho `apns-push-type` do APNs para a mensagem como `background`. Quando o Amazon SNS detecta que o dicionário `aps` contém a chave `content-available` definida como `1` e não contém nenhuma outra chave que possa acionar interações do usuário, ele define o cabeçalho como `background`.

### Usando cargas do Google Firebase Cloud Messaging (FCM) v1 no Amazon SNS

O Amazon SNS oferece suporte ao uso da API HTTP v1 do FCM para enviar notificações para destinos Android, iOS e Webpush. Este tópico fornece exemplos da estrutura de carga útil ao publicar notificações push móveis usando a CLI ou a API do Amazon SNS.

Você pode incluir os seguintes tipos de mensagem em sua carga ao enviar uma notificação do FCM:

- Mensagem de dados — Uma mensagem de dados é gerenciada pelo seu aplicativo cliente e contém pares de valores-chave personalizados. Ao criar uma mensagem de dados, você deve incluir a chave `data` com um objeto JSON como valor e, em seguida, inserir seus pares de valores-chave personalizados.

- Mensagem de notificação ou mensagem de exibição — uma mensagem de notificação contém um conjunto predefinido de chaves gerenciadas pelo SDK do FCM. Essas chaves variam de acordo com o tipo de dispositivo para o qual você está entregando. Para obter mais informações sobre as chaves de notificação específicas da plataforma, consulte o seguinte:
  - [Chaves de notificação do Android](#)
  - [Chaves de notificação do APNS](#)
  - [Chaves de notificação Webpush](#)

Para obter mais informações sobre os tipos de mensagens do FCM, consulte [Tipos de mensagens](#) na documentação do Firebase do Google.

## Sumário

- [Usando a estrutura de carga útil do FCM v1 para enviar mensagens](#)
- [Usando a estrutura de carga útil legada para enviar mensagens para a API FCM v1](#)
- [Eventos de falha na entrega do FCM](#)

## Usando a estrutura de carga útil do FCM v1 para enviar mensagens

Se você estiver criando um aplicativo FCM pela primeira vez ou quiser aproveitar os recursos do FCM v1, você pode optar por enviar uma carga no formato FCM v1. Para fazer isso, você deve incluir a chave `fcmV1Message` de nível superior. Para obter mais informações sobre a criação de cargas do FCM v1, consulte [Migração de APIs legadas do FCM para HTTP v1](#) e [Personalização de uma mensagem em várias plataformas](#) na documentação do Firebase do Google.

Exemplo de carga útil do FCM v1 enviada para o Amazon SNS:

### Note

O valor da GCM chave usado no exemplo a seguir deve ser codificado como uma string ao publicar uma notificação usando o Amazon SNS.

```
{
  "GCM": "{
    \"fcmV1Message\": {
      \"validate_only\" : false,
      \"message\" :
```

```

{
  \"notification\": {
    \"title\": \"string\",
    \"body\": \"string\"
  },
  \"data\": {
    \"dataGen\": \"priority message\",
  },
  \"android\": {
    \"priority\": \"high\",
    \"notification\": {
      \"body_loc_args\": [
        \"string\"
      ],
      \"title_loc_args\": [
        \"string\"
      ],
      \"sound\": \"string\",
      \"title_loc_key\": \"string\",
      \"title\": \"string\",
      \"body\": \"string\",
      \"click_action\": \"clicky_clacky\",
      \"body_loc_key\": \"string\"
    },
    \"data\": {
      \"dataAndroid\": \"priority message\",
    },
    \"ttl\": \"10023.32s\"
  },
  \"apns\": {
    \"payload\": {
      \"aps\": {
        \"alert\": {
          \"subtitle\": \"string\",
          \"title-loc-args\": [
            \"string\"
          ],
          \"title-loc-key\": \"string\",
          \"loc-args\": [
            \"string\"
          ],
          \"loc-key\": \"string\",
          \"title\": \"string\",
          \"body\": \"string\"
        }
      }
    }
  }
}

```



- [Sobre as mensagens do FCM](#)
- [Recurso REST: projects.messages](#)

Usando a estrutura de carga útil legada para enviar mensagens para a API FCM v1

Ao migrar para o FCM v1, você não precisa alterar a estrutura de carga que estava usando para suas credenciais legadas. O Amazon SNS transforma sua carga na nova estrutura de carga útil do FCM v1 e envia para o Google.

Formato da carga útil da mensagem de entrada:

```
{
  "GCM": "{\"notification\": {\"title\": \"string\", \"body\": \"string\",
  \"android_channel_id\": \"string\", \"body_loc_args\": [\"string\"], \"body_loc_key\":
  \"string\", \"click_action\": \"string\", \"color\": \"string\", \"icon\": \"string
  \", \"sound\": \"string\", \"tag\": \"string\", \"title_loc_args\": [\"string\"],
  \"title_loc_key\": \"string\"}, \"data\": {\"message\": \"priority message\"}}"
```

Mensagem enviada ao Google:

```
{
  "message": {
    "token": "****",
    "notification": {
      "title": "string",
      "body": "string"
    },
    "android": {
      "priority": "high",
      "notification": {
        "body_loc_args": [
          "string"
        ],
        "title_loc_args": [
          "string"
        ],
        "color": "string",
        "sound": "string",
        "icon": "string",
        "tag": "string",
        "title_loc_key": "string",
```

```
    "title": "string",
    "body": "string",
    "click_action": "string",
    "channel_id": "string",
    "body_loc_key": "string"
  },
  "data": {
    "message": "priority message"
  }
},
"apns": {
  "payload": {
    "aps": {
      "alert": {
        "title-loc-args": [
          "string"
        ],
        "title-loc-key": "string",
        "loc-args": [
          "string"
        ],
        "loc-key": "string",
        "title": "string",
        "body": "string"
      },
      "category": "string",
      "sound": "string"
    }
  }
},
"webpush": {
  "notification": {
    "icon": "string",
    "tag": "string",
    "body": "string",
    "title": "string"
  },
  "data": {
    "message": "priority message"
  }
},
"data": {
  "message": "priority message"
}
```

```
}  
}
```

## Riscos potenciais

- O mapeamento legado para v1 não é compatível com o Apple Push Notification Service (APNS) headers nem com as `fcm_options` chaves. Se você quiser usar esses campos, envie uma carga útil do FCM v1.
- Em alguns casos, os cabeçalhos das mensagens são exigidos pelo FCM v1 para enviar notificações silenciosas aos seus dispositivos APNs. Se você estiver enviando notificações silenciosas para seus dispositivos APNs, elas não funcionarão com a abordagem antiga. Em vez disso, recomendamos usar a carga útil do FCM v1 para evitar problemas inesperados. Para encontrar uma lista de cabeçalhos de APNs e para que eles são usados, consulte [Comunicação com APNs](#) no Guia do Desenvolvedor da Apple.
- Se você estiver usando o atributo TTL Amazon SNS ao enviar sua notificação, ela só será atualizada no `android` campo. Se você quiser definir o atributo TTL APNS, use a carga útil do FCM v1.
- As webpush teclas `androidapns`, e serão mapeadas e preenchidas com todas as chaves relevantes fornecidas. Por exemplo, se você fornecer `title`, que é uma chave compartilhada entre as três plataformas, o mapeamento do FCM v1 preencherá as três plataformas com o título que você forneceu.
- Algumas chaves compartilhadas entre plataformas esperam diferentes tipos de valor. Por exemplo, a `badge` chave passada para `apns` espera um valor inteiro, enquanto a `badge` chave passada para webpush espera um valor String. Nos casos em que você fornecer a `badge` chave, o mapeamento do FCM v1 preencherá somente a chave para a qual você forneceu um valor válido.

## Eventos de falha na entrega do FCM

A tabela a seguir fornece o tipo de falha do Amazon SNS que corresponde aos códigos de erro/status recebidos do Google para solicitações de notificação do FCM v1. Todos os códigos de erro observados recebidos da API FCM v1 estão disponíveis CloudWatch quando você configura o [registro do status de entrega do seu aplicativo](#).

Código de erro/status do FCM	Tipo de falha do Amazon SNS	Mensagem de falha	Causa e mitigação
UNREGISTERED	InvalidPlatformToken	O token da plataforma associado ao endpoint não é válido.	O token do dispositivo anexado ao seu endpoint está obsoleto ou é inválido. O Amazon SNS desativou seu endpoint. Atualize o endpoint do Amazon SNS para o token de dispositivo mais novo.
INVALID_ARGUMENT	InvalidNotification	O corpo da notificação é inválido.	O token do dispositivo ou a carga útil da mensagem podem ser inválidos. Verifique se a carga útil da mensagem é válida. Se a carga útil da mensagem for válida, atualize o endpoint do Amazon SNS para o token de dispositivo mais novo.
SENDER_ID_MISMATCH	InvalidPlatformToken	O token da plataforma associado ao endpoint não é válido.	O aplicativo da plataforma associado ao token do dispositivo não tem permissão para enviar para o token do dispositivo. Verifique se você está usando as credenciais corretas do FCM em seu aplicativo da

Código de erro/status do FCM	Tipo de falha do Amazon SNS	Mensagem de falha	Causa e mitigação
			plataforma Amazon SNS.
UNAVAILABLE	DependencyUnavailable	A dependência não está disponível.	O FCM não conseguiu processar a solicitação a tempo. Todas as novas tentativas executadas pelo Amazon SNS falharam. Você pode armazenar essas mensagens em uma fila de cartas mortas (DLQ) e reconduzi-las posteriormente.
INTERNAL	UnexpectedFailure	Falha inesperada; entre em contato com a Amazon. Frase de falha [Erro interno].	O servidor FCM encontrou um erro ao tentar processar sua solicitação. Todas as novas tentativas executadas pelo Amazon SNS falharam. Você pode armazenar essas mensagens em uma fila de cartas mortas (DLQ) e reconduzi-las posteriormente.

Código de erro/status do FCM	Tipo de falha do Amazon SNS	Mensagem de falha	Causa e mitigação
THIRD_PARTY_AUTH_ERROR	InvalidCredentials	As credenciais do aplicativo da plataforma não são válidas.	Uma mensagem direcionada a um dispositivo iOS ou a um dispositivo Webpush não pôde ser enviada. Verifique se suas credenciais de desenvolvimento e produção são válidas.
QUOTA_EXCEEDED	Throttled	Solicitação limitada por [gcm].	Uma cota de taxa de mensagem, cota de taxa de mensagem de dispositivo ou cota de taxa de mensagem de tópico foi excedida. Para obter informações sobre como resolver esse problema, consulte <a href="#">ErrorCode</a> a documentação do Firebase do Google.

Código de erro/status do FCM	Tipo de falha do Amazon SNS	Mensagem de falha	Causa e mitigação
PERMISSION_DENIED	InvalidNotification	O corpo da notificação é inválido.	No caso de uma PERMISSION_DENIED exceção, o chamador (seu aplicativo FCM) não tem permissão para executar a operação especificada na carga. Navegue até o console do FCM e verifique se suas credenciais têm as ações de API necessárias ativadas.

## Atributos de aplicativo móvel

O Amazon Simple Notification Service (Amazon SNS) oferece suporte para o registro em log do status de entrega das mensagens de notificação por push. Após configurar os atributos da aplicação, as entradas de log serão enviadas ao CloudWatch Logs para mensagens enviadas do Amazon SNS para endpoints móveis. O registro de status de entrega de mensagens proporciona um melhor insight operacional, por exemplo:

- Saiba se uma mensagem de notificação por push foi entregue do Amazon SNS para o serviço de notificações por push.
- Identifique a resposta enviada do serviço de notificações por push para o Amazon SNS.
- Determinar o tempo de permanência da mensagem (o tempo entre o carimbo de data e hora da publicação e antes do envio para um serviço de notificações por push).

Para configurar os atributos da aplicação para o status de entrega de mensagens, você pode usar o AWS Management Console, os AWS Software Development Kits (SDKs) ou a API de consulta.

### Tópicos

- [Configurar atributos de status de entrega de mensagens com o AWS Management Console](#)
- [Exemplos de CloudWatch Logs do status de entrega de mensagens do Amazon SNS](#)
- [Configurar atributos de status de entrega de mensagens com AWS SDKs](#)
- [Códigos de resposta da plataforma](#)

## Configurar atributos de status de entrega de mensagens com o AWS Management Console

1. Faça login no [console do Amazon SNS](#).
2. No painel de navegação, aponte para Mobile (Dispositivos móveis), em seguida escolha Push notifications (Notificações por push).
3. Na seção Platform Applications (Aplicações de plataforma), escolha a aplicação que contém os endpoints para os quais você deseja receber o CloudWatch Logs.
4. Escolha Application Actions (Ações do aplicativo) e, em seguida, selecione Delivery status (Status de entrega).
5. Na caixa de diálogo Delivery Status (Status da entrega), escolha Create IAM Roles (Criar funções do IAM).

Você será redirecionado para o console do IAM.

6. Escolha Allow (Permitir) para conceder ao Amazon SNS o acesso de gravação para usar o CloudWatch Logs em seu nome.
7. Volte para a caixa de diálogo Delivery Status (Status de entrega) e insira um número no campo Percentage of Success to Sample (0-100) (Porcentagem de sucesso de amostragem (0-100)) para a porcentagem de mensagens bem-sucedidas enviadas para as quais você deseja receber CloudWatch Logs.

### Note

Depois de configurar os atributos da aplicação para o status da entrega de mensagens, todas as entregas de mensagens com falha geram CloudWatch Logs.

8. Finalmente, escolha Save Configuration (Salvar configuração). Agora, você será capaz de visualizar e analisar o CloudWatch Logs contendo o status de entrega de mensagens. Para obter mais informações sobre o uso do CloudWatch, consulte a [Documentação do CloudWatch](#).

## Exemplos de CloudWatch Logs do status de entrega de mensagens do Amazon SNS

Após a configuração dos atributos de status de entrega de mensagens para um endpoint de aplicação, o CloudWatch Logs será gerado. Exemplos de logs, no formato JSON, são exibidos da seguinte forma:

### SUCCESS

```
{
  "status": "SUCCESS",
  "notification": {
    "timestamp": "2015-01-26 23:07:39.54",
    "messageId": "9655abe4-6ed6-5734-89f7-e6a6a42de02a"
  },
  "delivery": {
    "statusCode": 200,
    "dwellTimeMs": 65,
    "token": "Examplei7fFachkJ1xj1qT64RaBkcGHochmf1VQAr9k-
IBJtKjp7fedYPzEwT_Pq3Tu0lroqro1cwWJUvgkcPPYcaXCpWmG3Bqn-
wiqIEzp5zZ7y_jsM0PKPxKhddCzx6paEsyay9Zn3D4wNUJb8m6HXrBf9dqaEw",
    "attempts": 1,
    "providerResponse": "{\"multicast_id\":5138139752481671853,\"success
\":1,\"failure\":0,\"canonical_ids\":0,\"results\":[{\"message_id\":
\"0:1422313659698010%d6ba8edff9fd7ecd\"}]}",
    "destination": "arn:aws:sns:us-east-2:111122223333:endpoint/FCM/FCMPushApp/
c23e42de-3699-3639-84dd-65f84474629d"
  }
}
```

### FAILURE

```
{
  "status": "FAILURE",
  "notification": {
    "timestamp": "2015-01-26 23:29:35.678",
    "messageId": "c3ad79b0-8996-550a-8bfa-24f05989898f"
  },
  "delivery": {
    "statusCode": 8,
    "dwellTimeMs": 1451,
    "token": "example29z6j5c4df46f80189c4c83fjcgf7f6257e98542d2jt3395kj73",
    "attempts": 1,
  }
}
```

```
"providerResponse": "NotificationErrorResponse(command=8, status=InvalidToken,
id=1, cause=null)",
"destination": "arn:aws:sns:us-east-2:111122223333:endpoint/APNS_SANDBOX/
APNSPushApp/986cb8a1-4f6b-34b1-9a1b-d9e9cb553944"
}
}
```

Para obter uma lista de códigos de resposta do serviço de notificação por push, consulte [Códigos de resposta da plataforma](#).

## Configurar atributos de status de entrega de mensagens com AWS SDKs

Os [AWS SDKs](#) fornecem APIs em várias linguagens para usar atributos de status de entrega de mensagens com o Amazon SNS.

O seguinte exemplo Java mostra como usar a API `SetPlatformApplicationAttributes` para configurar atributos de aplicativo para status de entrega de mensagens para as mensagens de notificação por push. Você pode usar os seguintes atributos para o status de entrega de mensagens: `SuccessFeedbackRoleArn`, `FailureFeedbackRoleArn` e `SuccessFeedbackSampleRate`. Os atributos `SuccessFeedbackRoleArn` e `FailureFeedbackRoleArn` são usados para conceder acesso de gravação ao Amazon SNS para usar o CloudWatch Logs em seu nome. O atributo `SuccessFeedbackSampleRate` é para especificar a porcentagem de taxa de amostra (0-100) de mensagens bem-sucedidas. Após configurar o atributo `FailureFeedbackRoleArn`, todas as entregas de mensagens com falha geram CloudWatch Logs.

```
SetPlatformApplicationAttributesRequest setPlatformApplicationAttributesRequest = new
SetPlatformApplicationAttributesRequest();
Map<String, String> attributes = new HashMap<>();
attributes.put("SuccessFeedbackRoleArn", "arn:aws:iam::111122223333:role/SNS_CWlogs");
attributes.put("FailureFeedbackRoleArn", "arn:aws:iam::111122223333:role/SNS_CWlogs");
attributes.put("SuccessFeedbackSampleRate", "5");
setPlatformApplicationAttributesRequest.withAttributes(attributes);
setPlatformApplicationAttributesRequest.setPlatformApplicationArn("arn:aws:sns:us-
west-2:111122223333:app/FCM/FCMPushApp");
sns.setPlatformApplicationAttributes(setPlatformApplicationAttributesRequest);
```

Para obter mais informações sobre o SDK para Java, consulte [Getting Started with the AWS SDK for Java](#) (Conceitos básicos do AWS SDK for Java).

## Códigos de resposta da plataforma

A seguir, uma lista de links para os códigos de resposta do serviço de notificação por push:

Serviço de notificação por push	Códigos de resposta
Amazon Device Messaging (ADM)	Consulte <a href="#">Response Format</a> (“Formato de resposta”) na documentação do ADM.
Apple Push Notification Service (APNs)	Consulte Resposta HTTP/2 de APNs em <a href="#">Comunicação com APNs</a> no Guia de programação de notificação local e remota.
Firebase Cloud Messaging (FCM)	Consulte <a href="#">Códigos de respostas de erros de mensagens downstream</a> na documentação do Firebase Cloud Messaging.
Microsoft Push Notification Service for Windows Phone (MPNS)	Consulte <a href="#">Códigos de resposta do Serviço de notificação por push para Windows Phone 8</a> na documentação de desenvolvimento do Windows 8.
Windows Push Notification Services (WNS)	Consulte "Códigos de resposta" em <a href="#">Solicitação e cabeçalhos de resposta do Serviço de notificação por push (Aplicativos de tempo de execução do Windows)</a> na documentação de desenvolvimento do Windows 8.

## Eventos de aplicativo móvel

O Amazon SNS fornece suporte para acionar as notificações quando determinados eventos da aplicação ocorrem. Em seguida, você pode executar algumas ações programáticas nesse evento. A aplicação deve incluir suporte a um serviço de notificação por push, como Apple Push Notification Service (APNs), Firebase Cloud Messaging (FCM) e Windows Push Notification Services (WNS). Você define notificações de eventos do aplicativo usando o console do Amazon SNS ou os AWS CLI e AWS SDKs.

### Tópicos

- [Eventos do aplicativo disponíveis](#)
- [Enviar notificações por push para dispositivos móveis](#)

## Eventos do aplicativo disponíveis

As notificações de eventos do aplicativo rastreiam quando endpoints de plataforma individuais são criados, excluídos e atualizados, bem como as falhas de entrega. A seguir estão os nomes de atributo para os eventos do aplicativo.

Nome do atributo	Trigger para a notificação
EventEndpointCreated	Um novo endpoint de plataforma é adicionado ao seu aplicativo.
EventEndpointDeleted	Qualquer endpoint de plataforma associado ao seu aplicativo é excluído.
EventEndpointUpdated	Qualquer um dos atributos dos endpoints de plataforma associados ao seu aplicativo é alterado.
EventDeliveryFailure	Uma entrega para qualquer um dos endpoints de plataforma associados ao seu aplicativo encontra uma falha permanente. <div data-bbox="505 1209 1507 1570" style="border: 1px solid #00a0e3; border-radius: 10px; padding: 10px; margin-top: 10px;"><p> <b>Note</b></p><p>Para rastrear falhas de entrega no aplicativo da plataforma, inscreva-se em eventos de status de entrega de mensagens para o aplicativo. Para obter mais informações, consulte <a href="#">Usar atributos de aplicativo do Amazon SNS para obter o status de entrega de mensagens</a>.</p></div>

Você pode associar qualquer atributo a um aplicativo que poderá então receber essas notificações de eventos.

## Enviar notificações por push para dispositivos móveis

Para enviar notificações de eventos do aplicativo, você especifica um tópico para receber notificações de cada tipo de evento. Como o Amazon SNS envia notificações, o tópico pode direcioná-las para endpoints que executarão ações programáticas.

### Important

Aplicativos de grande volume criarão um grande número de notificações de eventos do aplicativo (por exemplo, dezenas de milhares), que sobrecarregarão os endpoints destinados ao uso humano, como endereços de e-mail, números de telefone e aplicativos móveis. Considere as seguintes diretrizes ao enviar as notificações de eventos do aplicativo para um tópico:

- Cada tópico que recebe notificações deve conter somente assinaturas de endpoints programáticos, como endpoints HTTP ou HTTPS, filas ou funções do Amazon SQS. AWS Lambda
- Para reduzir a quantidade de processamento que é acionada pelas notificações, limite as inscrições de cada tópico em um pequeno número (por exemplo, cinco ou menos).

Você pode enviar notificações de eventos do aplicativo usando o console do Amazon SNS, o AWS Command Line Interface (AWS CLI) ou os AWS SDKs.

### AWS Management Console

1. Faça login no [console do Amazon SNS](#).
2. No painel de navegação, escolha Mobile (Dispositivos móveis), Push notifications (Notificações por push).
3. Na página de notificações push móveis, na seção Aplicativos da plataforma, escolha um aplicativo e escolha Editar.
4. Expanda a seção Event notifications (Notificações de eventos).
5. Escolha Actions (Ações), Configure events (Configurar eventos).
6. Insira os ARNs dos tópicos a serem usados para os seguintes eventos:
  - Endpoint criado
  - Endpoint excluído

- Endpoint atualizado
- Falha na entrega

7. Escolha Salvar alterações.

## AWS CLI

Execute o comando [set-platform-application-attributes](#).

O exemplo a seguir define o mesmo tópico do Amazon SNS para todos os quatro eventos da aplicação:

```
aws sns set-platform-application-attributes
--platform-application-arn arn:aws:sns:us-east-1:12345EXAMPLE:app/FCM/
MyFCMPlatformApplication
--attributes EventEndpointCreated="arn:aws:sns:us-
east-1:12345EXAMPLE:MyFCMPlatformApplicationEvents",
EventEndpointDeleted="arn:aws:sns:us-
east-1:12345EXAMPLE:MyFCMPlatformApplicationEvents",
EventEndpointUpdated="arn:aws:sns:us-
east-1:12345EXAMPLE:MyFCMPlatformApplicationEvents",
EventDeliveryFailure="arn:aws:sns:us-
east-1:12345EXAMPLE:MyFCMPlatformApplicationEvents"
```

## AWS SDKs

Defina notificações de eventos do aplicativo enviando uma `SetPlatformApplicationAttributes` solicitação com a API do Amazon SNS usando AWS um SDK.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, incluindo ajuda para começar e informações sobre versões anteriores, consulte [Usando o Amazon SNS com um SDK AWS](#).

## Ações da API de push para dispositivos móveis

Para usar as APIs de push para dispositivos móveis do Amazon SNS, primeiro é necessário atender aos pré-requisitos do serviço de notificações por push, como o Apple Push Notification Service (APNs) e o Firebase Cloud Messaging (FCM). Para obter mais informações sobre os pré-requisitos, consulte [Pré-requisitos para notificações ao usuário do Amazon SNS](#).

Para enviar uma mensagem de notificação por push para um dispositivo um aplicativo móvel usando as APIs, você deve primeiro usar a ação `CreatePlatformApplication`, que retorna um atributo `PlatformApplicationArn`. O atributo `PlatformApplicationArn` é, então, usado por `CreatePlatformEndpoint`, que retorna um atributo `EndpointArn`. Em seguida, você pode usar o atributo `EndpointArn` com a ação `Publish` para enviar uma mensagem de notificação para um dispositivo e aplicativo móvel, ou você pode usar o atributo `EndpointArn` com a ação `Subscribe` de inscrição em um tópico. Para obter mais informações, consulte [. Visão geral do processo de notificação ao usuário.](#)

As APIs de push para dispositivos móveis do Amazon SNS são as seguintes:

### [CreatePlatformApplication](#)

Cria um objeto de aplicação da plataforma para um dos serviços de notificações por push compatíveis, como o APNs e o FCM, nos quais os dispositivos e as aplicações móveis podem se registrar. Retorna um atributo `PlatformApplicationArn`, que é usado pela ação `CreatePlatformEndpoint`.

### [CreatePlatformEndpoint](#)

Cria um endpoint para um dispositivo e aplicativo móvel em um dos serviços de notificação por push compatíveis. O `CreatePlatformEndpoint` usa o atributo `PlatformApplicationArn` retornado da ação `CreatePlatformApplication`. O atributo `EndpointArn`, que é retornado ao usar `CreatePlatformEndpoint`, é, então, usado com a ação `Publish` para enviar uma mensagem de notificação para um dispositivo e aplicativo móvel.

### [CreateTopic](#)

Cria um tópico no qual as mensagens podem ser publicadas.

### [DeleteEndpoint](#)

Exclui o endpoint para um dispositivo e aplicativo móvel em um dos serviços de notificações por push compatíveis.

### [DeletePlatformApplication](#)

Exclui um objeto de aplicativo de plataforma.

### [DeleteTopic](#)

Exclui um tópico e todas as suas inscrições.

### [GetEndpointAttributes](#)

Recupera os atributos do endpoint para um dispositivo e aplicativo móvel.

### [GetPlatformApplicationAttributes](#)

Recupera os atributos da plataforma de objeto de aplicativo.

### [ListEndpointsByPlatformApplication](#)

Lista os endpoints e atributos de endpoint para dispositivos e aplicativos móveis em um serviço de notificações por push compatível.

### [ListPlatformApplications](#)

Lista os objetos do aplicativo da plataforma para os serviços de notificações por push compatíveis.

### [Publish](#)

Envia uma mensagem de notificação para todos os endpoints inscritos em um tópico.

### [SetEndpointAttributes](#)

Define os atributos para um endpoint para um dispositivo e aplicativo móvel.

### [SetPlatformApplicationAttributes](#)

Define os atributos do objeto de aplicativo de plataforma.

### [Subscribe](#)

Prepara para assinar um endpoint enviando ao endpoint uma mensagem de confirmação. Para criar de fato uma inscrição, o proprietário do endpoint deve chamar a ação ConfirmSubscription com o token da mensagem de confirmação.

### [Unsubscribe](#)

Exclui a inscrição.

## Erros de API de push para dispositivos móveis

Os erros retornados pelas APIs do Amazon SNS para push para dispositivos móveis são listados na tabela a seguir. Para obter mais informações sobre as APIs do Amazon SNS para push para dispositivos móveis, consulte [Ações da API de push para dispositivos móveis](#).

Erro	Descrição	Código de status HTTPS	Ação API
Nome do aplicativo é uma string nula	O nome do aplicativo necessário está definido como nulo.	400	CreatePlatformApplication
Nome da plataforma é uma string nula	O nome da plataforma necessária está definido como nulo.	400	CreatePlatformApplication
Nome da plataforma é inválido	Um valor inválido ou fora do intervalo foi fornecido para o nome da plataforma.	400	CreatePlatformApplication
APNs: o principal não é um certificado válido	Um certificado inválido foi fornecido para o principal do APNs, que é o certificado SSL. Para obter mais informações, consulte <a href="#">CreatePlatformApplication</a> na Referência de API do Amazon Simple Notification Service.	400	CreatePlatformApplication
APNs: o principal é um certificado válido, mas não está no formato .pem	Um certificado válido que não está no formato .pem foi fornecido para o principal do APNs, que é o certificado SSL.	400	CreatePlatformApplication

Erro	Descrição	Código de status HTTPS	Ação API
APNs: o principal é um certificado expirado	Um certificado expirado foi fornecido para o principal do APNs, que é o certificado SSL.	400	CreatePlatformApplication
APNs: o principal não é um certificado emitido pela Apple	Um certificado não emitido pela Apple foi fornecido para o principal do APN, que é o certificado SSL.	400	CreatePlatformApplication
APNs: o principal não foi fornecido	O principal do APNs, que é o certificado SSL, não foi fornecido.	400	CreatePlatformApplication
APNs: a credencial não foi fornecida	As credenciais do APNS, que é a chave privada, não foram fornecidas. Para obter mais informações, consulte <a href="#">CreatePlatformApplication</a> na Referência de API do Amazon Simple Notification Service.	400	CreatePlatformApplication
APNs: a credencial não está em um formato .pem válido	As credenciais do APNs, que são a chave privada, não estão em um formato .pem válido.	400	CreatePlatformApplication

Erro	Descrição	Código de status HTTPS	Ação API
FCM: serverAPIKey não foi fornecida	As credenciais do FCM, que são a chave de API, não foram fornecidas. Para obter mais informações, consulte <a href="#">CreatePlatformApplication</a> na Referência de API do Amazon Simple Notification Service.	400	CreatePlatformApplication
FCM: a serverAPIKey está vazia	As credenciais do FCM, que são a chave de API, estão vazias.	400	CreatePlatformApplication
FCM: a serverAPIKey é uma string nula	As credenciais do FCM, que são a chave de API, são nulas.	400	CreatePlatformApplication
FCM: a serverAPIKey é inválida	As credenciais do FCM, que são a chave de API, são inválidas.	400	CreatePlatformApplication
ADM: clientsecret não foi fornecido	O segredo do cliente necessário não foi fornecido.	400	CreatePlatformApplication
ADM: clientsecret é uma string nula	A string necessária para o segredo do cliente é nula.	400	CreatePlatformApplication

Erro	Descrição	Código de status HTTPS	Ação API
ADM: client_secret é uma string vazia	A string necessária para o segredo do cliente está vazia.	400	CreatePlatformApplication
ADM: client_secret não é válido	A string necessária para o segredo do cliente não é válida.	400	CreatePlatformApplication
ADM: client_id é uma string vazia	A string necessária para o ID do cliente está vazia.	400	CreatePlatformApplication
ADM: clientId não foi fornecido	A string necessária para o ID do cliente não foi fornecida.	400	CreatePlatformApplication
ADM: clientid é uma string nula	A string necessária para o ID do cliente é nula.	400	CreatePlatformApplication
ADM: client_id não é válido	A string necessária para o ID do cliente não é válida.	400	CreatePlatformApplication
EventEndpointCreated tem um formato ARN inválido	EventEndpointCreated tem um formato ARN inválido.	400	CreatePlatformApplication
EventEndpointDeleted tem um formato ARN inválido	EventEndpointDeleted tem um formato ARN inválido.	400	CreatePlatformApplication
EventEndpointUpdated tem um formato ARN inválido	EventEndpointUpdated tem um formato ARN inválido.	400	CreatePlatformApplication

Erro	Descrição	Código de status HTTPS	Ação API
EventDeliveryAttemptFailure tem um formato ARN inválido	EventDeliveryAttemptFailure tem um formato ARN inválido.	400	CreatePlatformApplication
EventDeliveryFailure tem um formato ARN inválido	EventDeliveryFailure tem um formato ARN inválido.	400	CreatePlatformApplication
EventEndpointCreated não é um tópico existente	EventEndpointCreated não é um tópico existente.	400	CreatePlatformApplication
EventEndpointDeleted não é um tópico existente	EventEndpointDeleted não é um tópico existente.	400	CreatePlatformApplication
EventEndpointUpdated não é um tópico existente	EventEndpointUpdated não é um tópico existente.	400	CreatePlatformApplication
EventDeliveryAttemptFailure não é um tópico existente	EventDeliveryAttemptFailure não é um tópico existente.	400	CreatePlatformApplication
EventDeliveryFailure não é um tópico existente	EventDeliveryFailure não é um tópico existente.	400	CreatePlatformApplication
O ARN da plataforma é inválido	O ARN da plataforma é inválido.	400	SetPlatformAttributes
O ARN da plataforma é válido, mas não pertence ao usuário	O ARN da plataforma é válido, mas não pertence ao usuário.	400	SetPlatformAttributes

Erro	Descrição	Código de status HTTPS	Ação API
APNs: o principal não é um certificado válido	Um certificado inválido foi fornecido para o principal do APNs, que é o certificado SSL. Para obter mais informações, consulte <a href="#">CreatePlatformApplication</a> na Referência de API do Amazon Simple Notification Service.	400	SetPlatformAttributes
APNs: o principal é um certificado válido, mas não está no formato .pem	Um certificado válido que não está no formato .pem foi fornecido para o principal do APNs, que é o certificado SSL.	400	SetPlatformAttributes
APNs: o principal é um certificado expirado	Um certificado expirado foi fornecido para o principal do APNs, que é o certificado SSL.	400	SetPlatformAttributes
APNs: o principal não é um certificado emitido pela Apple	Um certificado não emitido pela Apple foi fornecido para o principal do APN, que é o certificado SSL.	400	SetPlatformAttributes

Erro	Descrição	Código de status HTTPS	Ação API
APNs: o principal não foi fornecido	O principal do APNs, que é o certificado SSL, não foi fornecido.	400	SetPlatformAttributes
APNs: a credencial não foi fornecida	As credenciais do APNS, que é a chave privada, não foram fornecidas. Para obter mais informações, consulte <a href="#">CreatePlatformApplication</a> na Referência de API do Amazon Simple Notification Service.	400	SetPlatformAttributes
APNs: a credencial não está em um formato .pem válido	As credenciais do APNs, que são a chave privada, não estão em um formato .pem válido.	400	SetPlatformAttributes
FCM: serverAPIKey não foi fornecida	As credenciais do FCM, que são a chave de API, não foram fornecidas. Para obter mais informações, consulte <a href="#">CreatePlatformApplication</a> na Referência de API do Amazon Simple Notification Service.	400	SetPlatformAttributes

Erro	Descrição	Código de status HTTPS	Ação API
FCM: a serverAPIKey é uma string nula	As credenciais do FCM, que são a chave de API, são nulas.	400	SetPlatformAttributes
ADM: clientId não foi fornecido	A string necessária para o ID do cliente não foi fornecida.	400	SetPlatformAttributes
ADM: clientId é uma string nula	A string necessária para o ID do cliente é nula.	400	SetPlatformAttributes
ADM: clientsecret não foi fornecido	O segredo do cliente necessário não foi fornecido.	400	SetPlatformAttributes
ADM: clientsecret é uma string nula	A string necessária para o segredo do cliente é nula.	400	SetPlatformAttributes
EventEndpointUpdated tem um formato ARN inválido	EventEndpointUpdated tem um formato ARN inválido.	400	SetPlatformAttributes
EventEndpointDeleted tem um formato ARN inválido	EventEndpointDeleted tem um formato ARN inválido.	400	SetPlatformAttributes
EventEndpointUpdated tem um formato ARN inválido	EventEndpointUpdated tem um formato ARN inválido.	400	SetPlatformAttributes
EventDeliveryAttemptFailure tem um formato ARN inválido	EventDeliveryAttemptFailure tem um formato ARN inválido.	400	SetPlatformAttributes

Erro	Descrição	Código de status HTTPS	Ação API
EventDeliveryFailure tem um formato ARN inválido	EventDeliveryFailure tem um formato ARN inválido.	400	SetPlatformAttributes
EventEndpointCreated não é um tópico existente	EventEndpointCreated não é um tópico existente.	400	SetPlatformAttributes
EventEndpointDeleted não é um tópico existente	EventEndpointDeleted não é um tópico existente.	400	SetPlatformAttributes
EventEndpointUpdated não é um tópico existente	EventEndpointUpdated não é um tópico existente.	400	SetPlatformAttributes
EventDeliveryAttemptFailure não é um tópico existente	EventDeliveryAttemptFailure não é um tópico existente.	400	SetPlatformAttributes
EventDeliveryFailure não é um tópico existente	EventDeliveryFailure não é um tópico existente.	400	SetPlatformAttributes
O ARN da plataforma é inválido	O ARN da plataforma é inválido.	400	GetPlatformApplicationAttributes
O ARN da plataforma é válido, mas não pertence ao usuário	O ARN da plataforma é válido, mas não pertence ao usuário.	403	GetPlatformApplicationAttributes
O token especificado é inválido	O token especificado é inválido.	400	ListPlatformApplications

Erro	Descrição	Código de status HTTPS	Ação API
O ARN da plataforma é inválido	O ARN da plataforma é inválido.	400	ListEndpointsByPlatformApplication
O ARN da plataforma é válido, mas não pertence ao usuário	O ARN da plataforma é válido, mas não pertence ao usuário.	404	ListEndpointsByPlatformApplication
O token especificado é inválido	O token especificado é inválido.	400	ListEndpointsByPlatformApplication
O ARN da plataforma é inválido	O ARN da plataforma é inválido.	400	DeletePlatformApplication
O ARN da plataforma é válido, mas não pertence ao usuário	O ARN da plataforma é válido, mas não pertence ao usuário.	403	DeletePlatformApplication
O ARN da plataforma é inválido	O ARN da plataforma é inválido.	400	CreatePlatformEndpoint
O ARN da plataforma é válido, mas não pertence ao usuário	O ARN da plataforma é válido, mas não pertence ao usuário.	404	CreatePlatformEndpoint
O token não foi especificado	O token não foi especificado.	400	CreatePlatformEndpoint
O token não é do tamanho certo	O token não tem o tamanho certo.	400	CreatePlatformEndpoint

Erro	Descrição	Código de status HTTPS	Ação API
Os dados do usuário do cliente são muito grandes	Os dados do usuário do cliente não podem ter mais de 2048 bytes em codificação UTF-8.	400	CreatePlatformEndpoint
O ARN do endpoint é inválido	O ARN do endpoint é inválido.	400	DeleteEndpoint
O ARN do endpoint é válido, mas não pertence ao usuário	O ARN do endpoint é válido, mas não pertence ao usuário.	403	DeleteEndpoint
O ARN do endpoint é inválido	O ARN do endpoint é inválido.	400	SetEndpointAttributes
O ARN do endpoint é válido, mas não pertence ao usuário	O ARN do endpoint é válido, mas não pertence ao usuário.	403	SetEndpointAttributes
O token não foi especificado	O token não foi especificado.	400	SetEndpointAttributes
O token não é do tamanho certo	O token não tem o tamanho certo.	400	SetEndpointAttributes
Os dados do usuário do cliente são muito grandes	Os dados do usuário do cliente não podem ter mais de 2048 bytes em codificação UTF-8.	400	SetEndpointAttributes
O ARN do endpoint é inválido	O ARN do endpoint é inválido.	400	GetEndpointAttributes

Erro	Descrição	Código de status HTTPS	Ação API
O ARN do endpoint é válido, mas não pertence ao usuário	O ARN do endpoint é válido, mas não pertence ao usuário.	403	GetEndpointAttributes
O ARN de destino é inválido	O ARN de destino é inválido.	400	Publish
O ARN de destino é válido, mas não pertence ao usuário	O ARN de destino é válido, mas não pertence ao usuário.	403	Publish
O formato da mensagem é inválido	O formato da mensagem é inválido.	400	Publish
O tamanho da mensagem é maior do que o comportado pelo protocolo/serviço final	O tamanho da mensagem é maior do que o comportado pelo protocolo/serviço final.	400	Publish

## Usar atributo de mensagem Time to Live (TTL) do Amazon SNS para notificações por push para dispositivos móveis

O Amazon Simple Notification Service (Amazon SNS) oferece suporte para configurar um atributo de mensagem Time to Live (TTL) para mensagens de notificação por push para dispositivos móveis. Isso se soma à capacidade existente de definir TTL no corpo da mensagem do Amazon SNS para os serviços móveis de notificação push que oferecem suporte a isso, como Amazon Device Messaging (ADM) e Firebase Cloud Messaging (FCM) ao enviar para Android.

O atributo de mensagem do TTL é usado para especificar os metadados de expiração sobre uma mensagem. Isso permite que você especifique a quantidade de tempo que o serviço de notificações por push, como o Apple Push Notification Service (APNs) ou o FCM, tem para entregar a mensagem ao endpoint. Se por algum motivo (por exemplo, se o dispositivo móvel foi desativado) a mensagem não for entregue no material do TTL especificado, ela será descartada e não serão feitas mais

tentativas de entrega. Para especificar o TTL nos atributos da mensagem, você pode usar os AWS Management Console kits de desenvolvimento de AWS software (SDKs) ou a API de consulta.

## Tópicos

- [Atributos da mensagem do TTL para serviços de notificação por push](#)
- [Ordem de precedência para determinar o TTL](#)
- [Especificando TTL usando o AWS Management Console](#)

## Atributos da mensagem do TTL para serviços de notificação por push

Veja a seguir uma lista dos atributos de mensagem TTL para serviços de notificação push que você pode usar para definir ao usar os AWS SDKs ou a API de consulta:

Serviço de notificação por push	Atributo de mensagem do TTL
Amazon Device Messaging (ADM)	<code>AWS.SNS.MOBILE.ADM.TTL</code>
Apple Push Notification Service (APNs)	<code>AWS.SNS.MOBILE.APNS.TTL</code>
Apple Push Notification Service Sandbox (APNs_SANDBOX)	<code>AWS.SNS.MOBILE.APNS_SANDBOX.TTL</code>
Baidu Cloud Push (Baidu)	<code>AWS.SNS.MOBILE.BAIDU.TTL</code>
Firebase Cloud Messaging (FCM ao enviar para Android)	<code>AWS.SNS.MOBILE.FCM.TTL</code>
Windows Push Notification Services (WNS)	<code>AWS.SNS.MOBILE.WNS.TTL</code>

Cada um dos serviços de notificação por push processa o TTL de maneira diferente. Com o Amazon SNS você tem uma visão abstrata do TTL em todos os serviços de notificação por push, o que facilita especificar o TTL. Ao usar o AWS Management Console para especificar o TTL (em segundos), você só precisa inserir o valor do TTL uma vez e o Amazon SNS calculará o TTL para cada um dos serviços de notificação push selecionados ao publicar a mensagem.

O TTL é relativo ao momento da publicação. Antes de enviar uma mensagem de notificação por push para um serviço de notificações por push específico, o Amazon SNS calcula o tempo de

permanência (o tempo entre o carimbo de data e hora da publicação e antes do envio para um serviço de notificações por push) para as notificações por push e passa o TTL restante para o serviço de notificações por push específico. Se o TTL for menor do que o tempo de permanência, o Amazon SNS não tentará publicar.

Se você especificar um TTL para uma mensagem de notificação por push, o valor do TTL deverá ser um número inteiro positivo, a menos que o valor de 0 tenha um significado específico para o serviço de notificação por push, como com APNs e FCM (ao enviar para Android). Se o valor do TTL for definido como 0 e o serviço de notificação por push não tiver um significado específico para 0, o Amazon SNS descartará a mensagem. Para obter mais informações sobre o parâmetro TTL definido como 0 ao usar o APNs, consulte [Table A-3 Item identifiers for remote notifications](#) (Tabela A-3 Identificadores de itens para notificações remotas) na documentação da [API Binary Provider](#).

## Ordem de precedência para determinar o TTL

A precedência que o Amazon SNS usa para determinar o TTL para uma mensagem de notificação por push é baseada na seguinte ordem, em que o número mais baixo tem prioridade mais alta:

1. TTL do atributo de mensagem
2. TTL do corpo da mensagem
3. TTL do serviço de notificação por push padrão (varia por serviço)
4. TTL padrão do Amazon SNS (4 semanas)

Se você definir valores diferentes de TTL (um em atributos de mensagem e outro no corpo da mensagem) para a mesma mensagem, o Amazon SNS modificará o TTL no corpo da mensagem de acordo com o TTL especificado no atributo de mensagem.

## Especificando TTL usando o AWS Management Console

1. Faça login no [console do Amazon SNS](#).
2. No painel de navegação, escolha Mobile (Dispositivos móveis), Push notifications (Notificações por push).
3. Na página Mobile push notifications (Notificações por push para dispositivos móveis), na seção Platform applications (Aplicativos de plataforma), selecione um aplicativo.
4. Na **MyApplication** página, na seção Endpoints, escolha um endpoint do aplicativo e escolha Publicar mensagem.

5. Na seção Message details (Detalhes da mensagem), insira o TTL (o número de segundos que o serviço de notificação por push tem para entregar a mensagem ao endpoint).
6. Selecione Publish message (Publicar mensagem).

## Regiões compatíveis com aplicações móveis

No momento, as aplicações móveis podem ser criadas apenas nas seguintes regiões:

- Leste dos EUA (Ohio)
- Leste dos EUA (N. da Virgínia)
- Oeste dos EUA (N. da Califórnia)
- Oeste dos EUA (Oregon)
- África (Cidade do Cabo)
- Ásia-Pacífico (Hong Kong)
- Ásia-Pacífico (Jacarta)
- Ásia-Pacífico (Mumbai)
- Asia Pacific (Osaka)
- Ásia-Pacífico (Seul)
- Ásia-Pacífico (Singapura)
- Ásia-Pacífico (Sydney)
- Ásia-Pacífico (Tóquio)
- Canadá (Central)
- Europa (Frankfurt)
- Europa (Irlanda)
- Europa (Londres)
- Europa (Milão)
- Europa (Paris)
- Europa (Estocolmo)
- Oriente Médio (Bahrein)
- Oriente Médio (Emirados Árabes Unidos)
- América do Sul (São Paulo)

- AWS GovCloud (Oeste dos EUA)

## Práticas recomendadas para notificações por push móveis

Esta seção descreve práticas recomendadas que podem ajudar você a melhorar seu envolvimento com os clientes.

### Gerenciamento de endpoints

Problemas de entrega podem ocorrer em situações em que os tokens de dispositivo mudam devido à ação de um usuário no dispositivo (por exemplo, um aplicativo é reinstalado no dispositivo) ou a [atualizações de certificado](#) que afetam os dispositivos que utilizam determinada versão do iOS. É uma prática recomendada pela Apple [registrar-se](#) com APNs toda vez que seu aplicativo é iniciado.

Como o token do dispositivo não muda sempre que um aplicativo é aberto por um usuário, é possível usar a API [CreatePlatformEndpoint](#) idempotente. No entanto, isso pode introduzir duplicatas para o mesmo dispositivo nos casos em que o token em si é inválido ou se o endpoint for válido, mas desativado (por exemplo, uma incompatibilidade de ambientes de produção e sandbox).

Um mecanismo de gerenciamento de tokens de dispositivo, como o contido no [pseudocódigo](#) pode ser usado.

Para obter informações sobre como gerenciar e manter os tokens do dispositivo FCM v1, consulte [Gerenciamento de endpoints do Firebase Cloud Messaging \(FCM\)](#)

### Registro em log do status de entrega

Para monitorar o status de entrega de notificações por push, recomendamos que você habilite o registro de status de entrega para a aplicação da plataforma Amazon SNS. Isso ajuda você a solucionar problemas de falhas de entrega porque os logs contêm [códigos de resposta](#) do provedor retornados do serviço de plataforma push. Para obter detalhes sobre como habilitar o registro de status de entrega, consulte [Como faço para acessar os logs de entrega de tópicos do Amazon SNS para notificações por push?](#)

### Notificações de eventos

Para gerenciar endpoints de forma orientada por eventos, você pode usar a funcionalidade [notificações de eventos](#). Isso permite que o tópico configurado do Amazon SNS faça fanout de eventos para os assinantes, como uma função do Lambda, para eventos de aplicações de plataforma de criação de endpoint, exclusão, atualizações e falhas de entrega.

## Notificações por e-mail

Esta página descreve como inscrever um [endereço de e-mail](#) em um tópico do Amazon SNS usando o AWS Management Console, AWS SDK for Java, ou AWS SDK for .NET

### Observações

- Não é possível personalizar o corpo da mensagem de e-mail. O recurso de entrega de e-mail destina-se a fornecer alertas internos do sistema, não mensagens de marketing.
- A assinatura direta de endpoints de e-mail só é compatível com tópicos-padrão.
- A taxa de transferência de entrega de e-mail é limitada de acordo com as [Cotas do Amazon SNS](#).

### Important

Para evitar que os destinatários da lista de e-mails cancelem a assinatura de todos os destinatários dos e-mails de tópicos do Amazon SNS, consulte [Configurar uma assinatura de e-mail que requer autenticação para cancelar a assinatura](#) no AWS Support.

## Para inscrever um endereço de e-mail em um tópico do Amazon SNS usando o AWS Management Console

1. Faça login no [console do Amazon SNS](#).
2. No painel de navegação à esquerda, escolha Assinaturas.
3. Na página Subscriptions (Assinaturas), escolha Create subscription (Criar assinatura).
4. Na página Create subscription (Criar inscrição), na seção Details (detalhes), faça o seguinte:
  - a. Em Topic ARN (ARN do tópico), escolha o nome do recurso da Amazon (ARN) de um tópico.
  - b. Em Protocolo, escolha Email.
  - c. Para Endpoint, insira o endereço de e-mail.

- d. (Opcional) Para configurar uma política de filtros, expanda a seção Subscription filter policy (Política de filtro de assinatura). Para ter mais informações, consulte [Políticas de filtro de assinatura do Amazon SNS](#).
- e. (Opcional) Para habilitar a filtragem baseada em carga útil, configure Filter Policy Scope como MessageBody. Para ter mais informações, consulte [Escopo de política de filtro de assinaturas do Amazon SNS](#).
- f. (Opcional) Para configurar uma fila de mensagens mortas para a assinatura, expanda a seção Redrive policy (dead-letter queue) (Política de redirecionamento (fila de mensagens mortas)). Para ter mais informações, consulte [Filas de mensagens mortas \(DLQs\) do Amazon SNS](#).
- g. Selecione Criar assinatura.

O console cria a assinatura e abre a página Details (Detalhes) da assinatura.

Você deve confirmar a assinatura para que o endereço de e-mail comece a receber mensagens.

Para confirmar uma assinatura

1. Verifique sua caixa de entrada de e-mail e escolha Confirm subscription (Confirmar a assinatura) no e-mail do Amazon SNS.
2. O Amazon SNS abre seu navegador da Web e exibe uma confirmação de assinatura com seu ID de assinatura.

## Para inscrever um endereço de e-mail em um tópico do Amazon SNS usando um SDK AWS

Para usar um AWS SDK, você deve configurá-lo com suas credenciais. Para obter mais informações, consulte [The shared config and credentials files](#) (“Os arquivos compartilhados de configuração e de credenciais”) no Guia de referência de AWS SDKs e ferramentas.

Os exemplos de código a seguir mostram como usar `Subscribe`.

## .NET

### AWS SDK for .NET

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

Inscreva um endereço de e-mail em um tópico.

```
/// <summary>
/// Creates a new subscription to a topic.
/// </summary>
/// <param name="client">The initialized Amazon SNS client object, used
/// to create an Amazon SNS subscription.</param>
/// <param name="topicArn">The ARN of the topic to subscribe to.</param>
/// <returns>A SubscribeResponse object which includes the subscription
/// ARN for the new subscription.</returns>
public static async Task<SubscribeResponse> TopicSubscribeAsync(
    IAmazonSimpleNotificationService client,
    string topicArn)
{
    SubscribeRequest request = new SubscribeRequest()
    {
        TopicArn = topicArn,
        ReturnSubscriptionArn = true,
        Protocol = "email",
        Endpoint = "recipient@example.com",
    };

    var response = await client.SubscribeAsync(request);

    return response;
}
```

Inscreva uma fila em um tópico com filtros opcionais.

```
/// <summary>
/// Subscribe a queue to a topic with optional filters.
/// </summary>
/// <param name="topicArn">The ARN of the topic.</param>
/// <param name="useFifoTopic">The optional filtering policy for the
subscription.</param>
/// <param name="queueArn">The ARN of the queue.</param>
/// <returns>The ARN of the new subscription.</returns>
public async Task<string> SubscribeTopicWithFilter(string topicArn, string?
filterPolicy, string queueArn)
{
    var subscribeRequest = new SubscribeRequest()
    {
        TopicArn = topicArn,
        Protocol = "sqs",
        Endpoint = queueArn
    };

    if (!string.IsNullOrEmpty(filterPolicy))
    {
        subscribeRequest.Attributes = new Dictionary<string, string>
{ { "FilterPolicy", filterPolicy } };
    }

    var subscribeResponse = await
_amazonSNSClient.SubscribeAsync(subscribeRequest);
    return subscribeResponse.SubscriptionArn;
}
```

- Para obter detalhes da API, consulte [Subscribe](#) na Referência da API do AWS SDK for .NET .

## C++

### SDK for C++

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

## Inscreva um endereço de e-mail em um tópico.

```
#!/ Subscribe to an Amazon Simple Notification Service (Amazon SNS) topic with
delivery to an email address.
/*!
 \param topicARN: An SNS topic Amazon Resource Name (ARN).
 \param emailAddress: An email address.
 \param clientConfiguration: AWS client configuration.
 \return bool: Function succeeded.
 */
bool AwsDoc::SNS::subscribeEmail(const Aws::String &topicARN,
                                const Aws::String &emailAddress,
                                const Aws::Client::ClientConfiguration
&clientConfiguration) {
    Aws::SNS::SNSClient snsClient(clientConfiguration);

    Aws::SNS::Model::SubscribeRequest request;
    request.SetTopicArn(topicARN);
    request.SetProtocol("email");
    request.SetEndpoint(emailAddress);

    const Aws::SNS::Model::SubscribeOutcome outcome =
snsClient.Subscribe(request);

    if (outcome.IsSuccess()) {
        std::cout << "Subscribed successfully." << std::endl;
        std::cout << "Subscription ARN " <<
outcome.GetResult().GetSubscriptionArn()
        << "." << std::endl;
    }
    else {
        std::cerr << "Error while subscribing " <<
outcome.GetError().GetMessage()
        << std::endl;
    }

    return outcome.IsSuccess();
}
```

## Inscreva um aplicativo móvel em um tópico.

```
#!/ Subscribe to an Amazon Simple Notification Service (Amazon SNS) topic with
delivery to a mobile app.
/*!
  \param topicARN: The Amazon Resource Name (ARN) for an Amazon SNS topic.
  \param endpointARN: The ARN for a mobile app or device endpoint.
  \param clientConfiguration: AWS client configuration.
  \return bool: Function succeeded.
*/
bool
AwsDoc::SNS::subscribeApp(const Aws::String &topicARN,
                          const Aws::String &endpointARN,
                          const Aws::Client::ClientConfiguration
&clientConfiguration) {
    Aws::SNS::SNSClient snsClient(clientConfiguration);

    Aws::SNS::Model::SubscribeRequest request;
    request.SetTopicArn(topicARN);
    request.SetProtocol("application");
    request.SetEndpoint(endpointARN);

    const Aws::SNS::Model::SubscribeOutcome outcome =
snsClient.Subscribe(request);

    if (outcome.IsSuccess()) {
        std::cout << "Subscribed successfully." << std::endl;
        std::cout << "Subscription ARN '" <<
outcome.GetResult().GetSubscriptionArn()
        << "'." << std::endl;
    }
    else {
        std::cerr << "Error while subscribing " <<
outcome.GetError().GetMessage()
        << std::endl;
    }

    return outcome.IsSuccess();
}
```

Inscreva uma função Lambda em um tópico.

```

//! Subscribe to an Amazon Simple Notification Service (Amazon SNS) topic with
delivery to an AWS Lambda function.
/*!
 \param topicARN: The Amazon Resource Name (ARN) for an Amazon SNS topic.
 \param lambdaFunctionARN: The ARN for an AWS Lambda function.
 \param clientConfiguration: AWS client configuration.
 \return bool: Function succeeded.
*/
bool AwsDoc::SNS::subscribeLambda(const Aws::String &topicARN,
                                  const Aws::String &lambdaFunctionARN,
                                  const Aws::Client::ClientConfiguration
&clientConfiguration) {
    Aws::SNS::SNSClient snsClient(clientConfiguration);

    Aws::SNS::Model::SubscribeRequest request;
    request.SetTopicArn(topicARN);
    request.SetProtocol("lambda");
    request.SetEndpoint(lambdaFunctionARN);

    const Aws::SNS::Model::SubscribeOutcome outcome =
snsClient.Subscribe(request);

    if (outcome.IsSuccess()) {
        std::cout << "Subscribed successfully." << std::endl;
        std::cout << "Subscription ARN '" <<
outcome.GetResult().GetSubscriptionArn()
        << "'." << std::endl;
    }
    else {
        std::cerr << "Error while subscribing " <<
outcome.GetError().GetMessage()
        << std::endl;
    }

    return outcome.IsSuccess();
}

```

Inscreva uma fila SQS em um tópico.

```

Aws::Client::ClientConfiguration clientConfig;
// Optional: Set to the AWS Region (overrides config file).
// clientConfig.region = "us-east-1";

```

```

    Aws::SNS::SNSClient snsClient(clientConfiguration);

    Aws::SNS::Model::SubscribeRequest request;
    request.SetTopicArn(topicARN);
    request.SetProtocol("sqs");
    request.SetEndpoint(queueARN);

    Aws::SNS::Model::SubscribeOutcome outcome =
snsClient.Subscribe(request);

    if (outcome.IsSuccess()) {
        Aws::String subscriptionARN =
outcome.GetResult().GetSubscriptionArn();
        std::cout << "The queue '" << queueName
            << "' has been subscribed to the topic '"
            << "'" << topicName << "'" << std::endl;
        std::cout << "with the subscription ARN '" << subscriptionARN <<
". "
            << std::endl;
        subscriptionARNS.push_back(subscriptionARN);
    }
    else {
        std::cerr << "Error with TopicsAndQueues::Subscribe. "
            << outcome.GetError().GetMessage()
            << std::endl;

        cleanUp(topicARN,
            queueURLS,
            subscriptionARNS,
            snsClient,
            sqsClient);

        return false;
    }
}

```

Inscreva-se com um filtro para um tópico.

```

static const Aws::String TONE_ATTRIBUTE("tone");
static const Aws::Vector<Aws::String> TONES = {"cheerful", "funny",
"serious",
"sincere"};

```

```
Aws::Client::ClientConfiguration clientConfig;
// Optional: Set to the AWS Region (overrides config file).
// clientConfig.region = "us-east-1";

Aws::SNS::SNSClient snsClient(clientConfiguration);

    Aws::SNS::Model::SubscribeRequest request;
    request.SetTopicArn(topicARN);
    request.SetProtocol("sqs");
    request.SetEndpoint(queueARN);
    if (isFifoTopic) {
        if (first) {
            std::cout << "Subscriptions to a FIFO topic can have
filters."
                        << std::endl;
            std::cout
                << "If you add a filter to this subscription, then
only the filtered messages "
                << "will be received in the queue." << std::endl;
            std::cout << "For information about message filtering, "
                << "see https://docs.aws.amazon.com/sns/latest/dg/
sns-message-filtering.html"
                << std::endl;
            std::cout << "For this example, you can filter messages by a
\""
                << TONE_ATTRIBUTE << "\" attribute." << std::endl;
        }

        std::ostringstream ostream;
        ostream << "Filter messages for \"" << queueName
                << "\"'s subscription to the topic \""
                << topicName << "\"? (y/n)";

        // Add filter if user answers yes.
        if (askYesNoQuestion(ostream.str())) {
            Aws::String jsonPolicy = getFilterPolicyFromUser();
            if (!jsonPolicy.empty()) {
                filteringMessages = true;

                std::cout << "This is the filter policy for this
subscription."
                        << std::endl;
                std::cout << jsonPolicy << std::endl;
            }
        }
    }
}
```

```

        request.AddAttributes("FilterPolicy", jsonPolicy);
    }
    else {
        std::cout
            << "Because you did not select any attributes, no
filter "
            << "will be added to this subscription." <<
std::endl;
    }
}
} // if (isFifoTopic)
Aws::SNS::Model::SubscribeOutcome outcome =
snsClient.Subscribe(request);

if (outcome.IsSuccess()) {
    Aws::String subscriptionARN =
outcome.GetResult().GetSubscriptionArn();
    std::cout << "The queue '" << queueName
        << "' has been subscribed to the topic '"
        << "'" << topicName << "'" << std::endl;
    std::cout << "with the subscription ARN '" << subscriptionARN <<
"."
        << std::endl;
    subscriptionARNS.push_back(subscriptionARN);
}
else {
    std::cerr << "Error with TopicsAndQueues::Subscribe. "
        << outcome.GetError().GetMessage()
        << std::endl;

    cleanUp(topicARN,
        queueURLS,
        subscriptionARNS,
        snsClient,
        sqsClient);

    return false;
}

//! Routine that lets the user select attributes for a subscription filter
policy.
/*!
\sa getFilterPolicyFromUser()

```

```
\return Aws::String: The filter policy as JSON.
*/
Aws::String AwsDoc::TopicsAndQueues::getFilterPolicyFromUser() {
    std::cout
        << "You can filter messages by one or more of the following \""
        << TONE_ATTRIBUTE << "\" attributes." << std::endl;

    std::vector<Aws::String> filterSelections;
    int selection;
    do {
        for (size_t j = 0; j < TONES.size(); ++j) {
            std::cout << " " << (j + 1) << ". " << TONES[j]
                << std::endl;
        }
        selection = askQuestionForIntRange(
            "Enter a number (or enter zero to stop adding more). ",
            0, static_cast<int>(TONES.size()));

        if (selection != 0) {
            const Aws::String &selectedTone(TONES[selection - 1]);
            // Add the tone to the selection if it is not already added.
            if (std::find(filterSelections.begin(),
                filterSelections.end(),
                selectedTone)
                == filterSelections.end()) {
                filterSelections.push_back(selectedTone);
            }
        }
    } while (selection != 0);

    Aws::String result;
    if (!filterSelections.empty()) {
        std::ostringstream jsonPolicyStream;
        jsonPolicyStream << "{ \"" << TONE_ATTRIBUTE << "\": [";

        for (size_t j = 0; j < filterSelections.size(); ++j) {
            jsonPolicyStream << "\"" << filterSelections[j] << "\"";
            if (j < filterSelections.size() - 1) {
                jsonPolicyStream << ",";
            }
        }
        jsonPolicyStream << "]" }";
    }
}
```

```
        result = jsonPolicyStream.str();
    }

    return result;
}
```

- Para obter detalhes da API, consulte [Subscribe](#) na Referência da API do AWS SDK for C++ .

## CLI

### AWS CLI

Para inscrever-se em um tópico

O comando `subscribe` a seguir inscreve um endereço de e-mail no tópico especificado.

```
aws sns subscribe \
  --topic-arn arn:aws:sns:us-west-2:123456789012:my-topic \
  --protocol email \
  --notification-endpoint my-email@example.com
```

Saída:

```
{
  "SubscriptionArn": "pending confirmation"
}
```

- Para obter detalhes da API, consulte [Subscribe](#) na Referência de comandos da AWS CLI .

## Go

### SDK for Go V2

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

Inscreva uma fila em um t3pico com filtros opcionais.

```
// SnsActions encapsulates the Amazon Simple Notification Service (Amazon SNS)
actions
// used in the examples.
type SnsActions struct {
    SnsClient *sns.Client
}

// SubscribeQueue subscribes an Amazon Simple Queue Service (Amazon SQS) queue to
an
// Amazon SNS topic. When filterMap is not nil, it is used to specify a filter
policy
// so that messages are only sent to the queue when the message has the specified
attributes.
func (actor SnsActions) SubscribeQueue(topicArn string, queueArn string,
filterMap map[string][]string) (string, error) {
    var subscriptionArn string
    var attributes map[string]string
    if filterMap != nil {
        filterBytes, err := json.Marshal(filterMap)
        if err != nil {
            log.Printf("Couldn't create filter policy, here's why: %v\n", err)
            return "", err
        }
        attributes = map[string]string{"FilterPolicy": string(filterBytes)}
    }
    output, err := actor.SnsClient.Subscribe(context.TODO(), &sns.SubscribeInput{
        Protocol:          aws.String("sqs"),
        TopicArn:          aws.String(topicArn),
        Attributes:        attributes,
        Endpoint:          aws.String(queueArn),
        ReturnSubscriptionArn: true,
    })
    if err != nil {
        log.Printf("Couldn't subscribe queue %v to topic %v. Here's why: %v\n",
            queueArn, topicArn, err)
    } else {
        subscriptionArn = *output.SubscriptionArn
    }
}
```

```
    return subscriptionArn, err
}
```

- Para obter detalhes da API, consulte [Subscribe](#) na Referência da API do AWS SDK for Go .

## Java

### SDK para Java 2.x

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

Inscreva um endereço de e-mail em um tópico.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.sns.SnsClient;
import software.amazon.awssdk.services.sns.model.SnsException;
import software.amazon.awssdk.services.sns.model.SubscribeRequest;
import software.amazon.awssdk.services.sns.model.SubscribeResponse;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
 * started.html
 */
public class SubscribeEmail {
    public static void main(String[] args) {
        final String usage = ""
            Usage:    <topicArn> <email>

        Where:
            topicArn - The ARN of the topic to subscribe.
            email - The email address to use.
```

```

        """;

    if (args.length != 2) {
        System.out.println(usage);
        System.exit(1);
    }

    String topicArn = args[0];
    String email = args[1];
    SnsClient snsClient = SnsClient.builder()
        .region(Region.US_EAST_1)
        .build();

    subEmail(snsClient, topicArn, email);
    snsClient.close();
}

public static void subEmail(SnsClient snsClient, String topicArn, String
email) {
    try {
        SubscribeRequest request = SubscribeRequest.builder()
            .protocol("email")
            .endpoint(email)
            .returnSubscriptionArn(true)
            .topicArn(topicArn)
            .build();

        SubscribeResponse result = snsClient.subscribe(request);
        System.out.println("Subscription ARN: " + result.subscriptionArn() +
"\n\n Status is "
            + result.sdkHttpResponse().statusCode());

    } catch (SnsException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}
}
}

```

Inscreva um endpoint HTTP em um tópico.

```
import software.amazon.awssdk.regions.Region;
```

```
import software.amazon.awssdk.services.sns.SnsClient;
import software.amazon.awssdk.services.sns.model.SnsException;
import software.amazon.awssdk.services.sns.model.SubscribeRequest;
import software.amazon.awssdk.services.sns.model.SubscribeResponse;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */
public class SubscribeHTTPS {
    public static void main(String[] args) {
        final String usage = ""

                Usage:    <topicArn> <url>

                Where:
                    topicArn - The ARN of the topic to subscribe.
                    url - The HTTPS endpoint that you want to receive
notifications.

                """;

        if (args.length < 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String topicArn = args[0];
        String url = args[1];
        SnsClient snsClient = SnsClient.builder()
            .region(Region.US_EAST_1)
            .build();

        subHTTPS(snsClient, topicArn, url);
        snsClient.close();
    }

    public static void subHTTPS(SnsClient snsClient, String topicArn, String url)
    {
        try {
```

```
        SubscribeRequest request = SubscribeRequest.builder()
            .protocol("https")
            .endpoint(url)
            .returnSubscriptionArn(true)
            .topicArn(topicArn)
            .build();

        SubscribeResponse result = snsClient.subscribe(request);
        System.out.println("Subscription ARN is " + result.subscriptionArn()
+ "\n\n Status is "
            + result.sdkHttpResponse().statusCode());

    } catch (SnsException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}
}
```

Inscreva uma função Lambda em um tópico.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.sns.SnsClient;
import software.amazon.awssdk.services.sns.model.SnsException;
import software.amazon.awssdk.services.sns.model.SubscribeRequest;
import software.amazon.awssdk.services.sns.model.SubscribeResponse;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */
public class SubscribeLambda {

    public static void main(String[] args) {

        final String usage = ""
```

```
Usage:    <topicArn> <lambdaArn>

Where:
  topicArn - The ARN of the topic to subscribe.
  lambdaArn - The ARN of an AWS Lambda function.
""";

if (args.length != 2) {
    System.out.println(usage);
    System.exit(1);
}

String topicArn = args[0];
String lambdaArn = args[1];
SnsClient snsClient = SnsClient.builder()
    .region(Region.US_EAST_1)
    .build();

String arnValue = subLambda(snsClient, topicArn, lambdaArn);
System.out.println("Subscription ARN: " + arnValue);
snsClient.close();
}

public static String subLambda(SnsClient snsClient, String topicArn, String
lambdaArn) {
    try {
        SubscribeRequest request = SubscribeRequest.builder()
            .protocol("lambda")
            .endpoint(lambdaArn)
            .returnSubscriptionArn(true)
            .topicArn(topicArn)
            .build();

        SubscribeResponse result = snsClient.subscribe(request);
        return result.subscriptionArn();

    } catch (SnsException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
    return "";
}
}
```

- Para obter detalhes da API, consulte [Subscribe](#) na Referência da API do AWS SDK for Java 2.x .

## JavaScript

### SDK para JavaScript (v3)

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

Crie o cliente em um módulo separado e exporte-o.

```
import { SNSClient } from "@aws-sdk/client-sns";

// The AWS Region can be provided here using the `region` property. If you leave
// it blank
// the SDK will default to the region set in your AWS config.
export const snsClient = new SNSClient({});
```

Importe o SDK e os módulos do cliente e chame a API.

```
import { SubscribeCommand } from "@aws-sdk/client-sns";
import { snsClient } from "../libs/snsClient.js";

/**
 * @param {string} topicArn - The ARN of the topic for which you wish to confirm
 * a subscription.
 * @param {string} emailAddress - The email address that is subscribed to the
 * topic.
 */
export const subscribeEmail = async (
  topicArn = "TOPIC_ARN",
  emailAddress = "user@me.com",
) => {
```

```
const response = await snsClient.send(
  new SubscribeCommand({
    Protocol: "email",
    TopicArn: topicArn,
    Endpoint: emailAddress,
  }),
);
console.log(response);
// {
//   '$metadata': {
//     httpStatusCode: 200,
//     requestId: 'c8e35bcd-b3c0-5940-9f66-06f6fcc108f0',
//     extendedRequestId: undefined,
//     cfId: undefined,
//     attempts: 1,
//     totalRetryDelay: 0
//   },
//   SubscriptionArn: 'pending confirmation'
// }
};
```

Inscreva um aplicativo móvel em um tópico.

```
import { SubscribeCommand } from "@aws-sdk/client-sns";
import { snsClient } from "../libs/snsClient.js";

/**
 * @param {string} topicArn - The ARN of the topic the subscriber is subscribing
 * to.
 * @param {string} endpoint - The Endpoint ARN of an application. This endpoint
 * is created
 *
 * when an application registers for notifications.
 */
export const subscribeApp = async (
  topicArn = "TOPIC_ARN",
  endpoint = "ENDPOINT",
) => {
  const response = await snsClient.send(
    new SubscribeCommand({
      Protocol: "application",
      TopicArn: topicArn,
      Endpoint: endpoint,
    })
  );
}
```

```

    }),
  );
  console.log(response);
  // {
  //   '$metadata': {
  //     httpStatusCode: 200,
  //     requestId: 'c8e35bcd-b3c0-5940-9f66-06f6fcc108f0',
  //     extendedRequestId: undefined,
  //     cfId: undefined,
  //     attempts: 1,
  //     totalRetryDelay: 0
  //   },
  //   SubscriptionArn: 'pending confirmation'
  // }
  return response;
};

```

Inscreva uma função Lambda em um tópico.

```

import { SubscribeCommand } from "@aws-sdk/client-sns";
import { snsClient } from "../libs/snsClient.js";

/**
 * @param {string} topicArn - The ARN of the topic the subscriber is subscribing
 * to.
 * @param {string} endpoint - The Endpoint ARN of and AWS Lambda function.
 */
export const subscribeLambda = async (
  topicArn = "TOPIC_ARN",
  endpoint = "ENDPOINT",
) => {
  const response = await snsClient.send(
    new SubscribeCommand({
      Protocol: "lambda",
      TopicArn: topicArn,
      Endpoint: endpoint,
    }),
  );
  console.log(response);
  // {
  //   '$metadata': {
  //     httpStatusCode: 200,

```

```
//   requestId: 'c8e35bcd-b3c0-5940-9f66-06f6fcc108f0',
//   extendedRequestId: undefined,
//   cfId: undefined,
//   attempts: 1,
//   totalRetryDelay: 0
// },
//   SubscriptionArn: 'pending confirmation'
// }
return response;
};
```

Inscreva uma fila SQS em um tópico.

```
import { SubscribeCommand, SNSClient } from "@aws-sdk/client-sns";

const client = new SNSClient({});

export const subscribeQueue = async (
  topicArn = "TOPIC_ARN",
  queueArn = "QUEUE_ARN",
) => {
  const command = new SubscribeCommand({
    TopicArn: topicArn,
    Protocol: "sqs",
    Endpoint: queueArn,
  });

  const response = await client.send(command);
  console.log(response);
  // {
  //   '$metadata': {
  //     httpStatusCode: 200,
  //     requestId: '931e13d9-5e2b-543f-8781-4e9e494c5ff2',
  //     extendedRequestId: undefined,
  //     cfId: undefined,
  //     attempts: 1,
  //     totalRetryDelay: 0
  //   },
  //   SubscriptionArn: 'arn:aws:sns:us-east-1:xxxxxxxxxxxx:subscribe-queue-
  test-430895:xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxxx'
  // }
  return response;
};
```

```
};
```

Inscreva-se com um filtro para um tópico.

```
import { SubscribeCommand, SNSClient } from "@aws-sdk/client-sns";

const client = new SNSClient({});

export const subscribeQueueFiltered = async (
  topicArn = "TOPIC_ARN",
  queueArn = "QUEUE_ARN",
) => {
  const command = new SubscribeCommand({
    TopicArn: topicArn,
    Protocol: "sqs",
    Endpoint: queueArn,
    Attributes: {
      // This subscription will only receive messages with the 'event' attribute
      // set to 'order_placed'.
      FilterPolicyScope: "MessageAttributes",
      FilterPolicy: JSON.stringify({
        event: ["order_placed"],
      }),
    },
  });

  const response = await client.send(command);
  console.log(response);
  // {
  //   '$metadata': {
  //     httpStatusCode: 200,
  //     requestId: '931e13d9-5e2b-543f-8781-4e9e494c5ff2',
  //     extendedRequestId: undefined,
  //     cfId: undefined,
  //     attempts: 1,
  //     totalRetryDelay: 0
  //   },
  //   SubscriptionArn: 'arn:aws:sns:us-east-1:xxxxxxxxxxxx:subscribe-queue-
  // test-430895:xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxxx'
  // }
  return response;
};
```

- Para obter mais informações, consulte o [Guia do desenvolvedor do AWS SDK for JavaScript](#).
- Para obter detalhes da API, consulte [Subscribe](#) na Referência da API do AWS SDK for JavaScript .

## Kotlin

### SDK for Kotlin

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

Inscreva um endereço de e-mail em um tópico.

```
suspend fun subEmail(topicArnVal: String, email: String): String {  
  
    val request = SubscribeRequest {  
        protocol = "email"  
        endpoint = email  
        returnSubscriptionArn = true  
        topicArn = topicArnVal  
    }  
  
    SnsClient { region = "us-east-1" }.use { snsClient ->  
        val result = snsClient.subscribe(request)  
        return result.subscriptionArn.toString()  
    }  
}
```

Inscreva uma função Lambda em um tópico.

```
suspend fun subLambda(topicArnVal: String?, lambdaArn: String?) {  
  
    val request = SubscribeRequest {
```

```
        protocol = "lambda"
        endpoint = lambdaArn
        returnSubscriptionArn = true
        topicArn = topicArnVal
    }

    SnsClient { region = "us-east-1" }.use { snsClient ->
        val result = snsClient.subscribe(request)
        println(" The subscription Arn is ${result.subscriptionArn}")
    }
}
```

- Para obter detalhes da API, consulte [Subscribe](#) na Referência da API do AWS SDK for Kotlin.

## PHP

### SDK para PHP

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

Inscreva um endereço de e-mail em um tópico.

```
require 'vendor/autoload.php';

use Aws\Exception\AwsException;
use Aws\Sns\SnsClient;

/**
 * Prepares to subscribe an endpoint by sending the endpoint a confirmation
 * message.
 *
 * This code expects that you have AWS credentials set up per:
 * https://docs.aws.amazon.com/sdk-for-php/v3/developer-guide/
 * guide_credentials.html
 */
```

```
$SnSClient = new SnsClient([
    'profile' => 'default',
    'region' => 'us-east-1',
    'version' => '2010-03-31'
]);

$protocol = 'email';
$endpoint = 'sample@example.com';
$topic = 'arn:aws:sns:us-east-1:111122223333:MyTopic';

try {
    $result = $SnSClient->subscribe([
        'Protocol' => $protocol,
        'Endpoint' => $endpoint,
        'ReturnSubscriptionArn' => true,
        'TopicArn' => $topic,
    ]);
    var_dump($result);
} catch (AwsException $e) {
    // output error message if fails
    error_log($e->getMessage());
}
```

Inscreva um endpoint HTTP em um tópico.

```
require 'vendor/autoload.php';

use Aws\Exception\AwsException;
use Aws\Sns\SnsClient;

/**
 * Prepares to subscribe an endpoint by sending the endpoint a confirmation
 * message.
 *
 * This code expects that you have AWS credentials set up per:
 * https://docs.aws.amazon.com/sdk-for-php/v3/developer-guide/
 * guide_credentials.html
 */
```

```
$SnSClient = new SnsClient([
    'profile' => 'default',
    'region' => 'us-east-1',
    'version' => '2010-03-31'
]);

$protocol = 'https';
$endpoint = 'https://';
$topic = 'arn:aws:sns:us-east-1:111122223333:MyTopic';

try {
    $result = $SnSClient->subscribe([
        'Protocol' => $protocol,
        'Endpoint' => $endpoint,
        'ReturnSubscriptionArn' => true,
        'TopicArn' => $topic,
    ]);
    var_dump($result);
} catch (AwsException $e) {
    // output error message if fails
    error_log($e->getMessage());
}
```

- Para obter detalhes da API, consulte [Subscribe](#) na Referência da API do AWS SDK for PHP

## Python

### SDK para Python (Boto3)

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

Inscreva um endereço de e-mail em um tópico.

```
class SnsWrapper:
    """Encapsulates Amazon SNS topic and subscription functions."""
```

```
def __init__(self, sns_resource):
    """
    :param sns_resource: A Boto3 Amazon SNS resource.
    """
    self.sns_resource = sns_resource

    @staticmethod
    def subscribe(topic, protocol, endpoint):
        """
        Subscribes an endpoint to the topic. Some endpoint types, such as email,
        must be confirmed before their subscriptions are active. When a
        subscription
        is not confirmed, its Amazon Resource Number (ARN) is set to
        'PendingConfirmation'.

        :param topic: The topic to subscribe to.
        :param protocol: The protocol of the endpoint, such as 'sms' or 'email'.
        :param endpoint: The endpoint that receives messages, such as a phone
        number
                        (in E.164 format) for SMS messages, or an email address
        for
                        email messages.
        :return: The newly added subscription.
        """
        try:
            subscription = topic.subscribe(
                Protocol=protocol, Endpoint=endpoint, ReturnSubscriptionArn=True
            )
            logger.info("Subscribed %s %s to topic %s.", protocol, endpoint,
                topic.arn)
        except ClientError:
            logger.exception(
                "Couldn't subscribe %s %s to topic %s.", protocol, endpoint,
                topic.arn
            )
            raise
        else:
            return subscription
```

- Para obter detalhes da API, consulte [Subscribe](#) na Referência da API do AWS SDK for Python (Boto3).

## Ruby

### SDK for Ruby

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

Inscreva um endereço de e-mail em um tópico.

```
require "aws-sdk-sns"
require "logger"

# Represents a service for creating subscriptions in Amazon Simple Notification
# Service (SNS)
class SubscriptionService
  # Initializes the SubscriptionService with an SNS client
  #
  # @param sns_client [Aws::SNS::Client] The SNS client
  def initialize(sns_client)
    @sns_client = sns_client
    @logger = Logger.new($stdout)
  end

  # Attempts to create a subscription to a topic
  #
  # @param topic_arn [String] The ARN of the SNS topic
  # @param protocol [String] The subscription protocol (e.g., email)
  # @param endpoint [String] The endpoint that receives the notifications (email
  # address)
  # @return [Boolean] true if subscription was successfully created, false
  # otherwise
  def create_subscription(topic_arn, protocol, endpoint)
    @sns_client.subscribe(topic_arn: topic_arn, protocol: protocol, endpoint:
    endpoint)
    @logger.info("Subscription created successfully.")
    true
  end
end
```

```
rescue Aws::SNS::Errors::ServiceError => e
  @logger.error("Error while creating the subscription: #{e.message}")
  false
end
end

# Main execution if the script is run directly
if $PROGRAM_NAME == __FILE__
  protocol = "email"
  endpoint = "EMAIL_ADDRESS" # Should be replaced with a real email address
  topic_arn = "TOPIC_ARN"    # Should be replaced with a real topic ARN

  sns_client = Aws::SNS::Client.new
  subscription_service = SubscriptionService.new(sns_client)

  @logger.info("Creating the subscription.")
  unless subscription_service.create_subscription(topic_arn, protocol, endpoint)
    @logger.error("Subscription creation failed. Stopping program.")
    exit 1
  end
end
end
```

- Para obter mais informações, consulte o [Guia do desenvolvedor do AWS SDK for Ruby](#).
- Para obter detalhes da API, consulte [Subscribe](#) na Referência da API do AWS SDK for Ruby .

## Rust

### SDK for Rust

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

Inscreva um endereço de e-mail em um tópico.

```
async fn subscribe_and_publish(
  client: &Client,
```

```
    topic_arn: &str,
    email_address: &str,
) -> Result<(), Error> {
    println!("Receiving on topic with ARN: `{}`", topic_arn);

    let rsp = client
        .subscribe()
        .topic_arn(topic_arn)
        .protocol("email")
        .endpoint(email_address)
        .send()
        .await?;

    println!("Added a subscription: {:?}", rsp);

    let rsp = client
        .publish()
        .topic_arn(topic_arn)
        .message("hello sns!")
        .send()
        .await?;

    println!("Published message: {:?}", rsp);

    Ok(())
}
```

- Para obter detalhes da API, consulte [Subscribe](#) na Referência da API do AWS SDK for Rust.

## SAP ABAP

### SDK para SAP ABAP

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

Inscreva um endereço de e-mail em um tópico.

```
TRY.  
    oo_result = lo_sns->subscribe(                                "oo_result is  
returned for testing purposes."  
        iv_topicarn = iv_topic_arn  
        iv_protocol = 'email'  
        iv_endpoint = iv_email_address  
        iv_returnsubscriptionarn = abap_true  
    ).  
    MESSAGE 'Email address subscribed to SNS topic.' TYPE 'I'.  
    CATCH /aws1/cx_snsnotfoundexception.  
        MESSAGE 'Topic does not exist.' TYPE 'E'.  
    CATCH /aws1/cx_snssubscriptionlmt00.  
        MESSAGE 'Unable to create subscriptions. You have reached the maximum  
number of subscriptions allowed.' TYPE 'E'.  
ENDTRY.
```

- Para obter detalhes da API, consulte [Inscrever-se](#) na Referência da API do AWS SDK para SAP ABAP.

# Exemplos de código para Amazon SNS usando SDKs AWS

Os exemplos de código a seguir mostram como usar o Amazon SNS com um kit de desenvolvimento AWS de software (SDK).

Ações são trechos de código de programas maiores e devem ser executadas em contexto. Embora as ações mostrem como chamar funções de serviço específicas, é possível ver as ações contextualizadas em seus devidos cenários e exemplos entre serviços.

Cenários são exemplos de código que mostram como realizar uma tarefa específica chamando várias funções dentro do mesmo serviço.

Exemplos entre serviços são amostras de aplicações que funcionam em vários Serviços da AWS.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulte [Usando o Amazon SNS com um SDK AWS](#). Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Conceitos básicos

## Olá, Amazon SNS

Os exemplos de código a seguir mostram como começar a usar o Amazon SNS.

.NET

AWS SDK for .NET

### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
using Amazon.SimpleNotificationService;
using Amazon.SimpleNotificationService.Model;

namespace SNSActions;

public static class HelloSNS
```

```
{
    static async Task Main(string[] args)
    {
        var snsClient = new AmazonSimpleNotificationServiceClient();

        Console.WriteLine($"Hello Amazon SNS! Following are some of your
topics:");
        Console.WriteLine();

        // You can use await and any of the async methods to get a response.
        // Let's get a list of topics.
        var response = await snsClient.ListTopicsAsync(
            new ListTopicsRequest());

        foreach (var topic in response.Topics)
        {
            Console.WriteLine($"\\tTopic ARN: {topic.TopicArn}");
            Console.WriteLine();
        }
    }
}
```

- Para obter detalhes da API, consulte [ListTopics](#) na Referência AWS SDK for .NET da API.

## C++

### SDK para C++

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

Código para o arquivo CMake CMakeLists.txt.

```
# Set the minimum required version of CMake for this project.
cmake_minimum_required(VERSION 3.13)

# Set the AWS service components used by this project.
set(SERVICE_COMPONENTS sns)
```

```
# Set this project's name.
project("hello_sns")

# Set the C++ standard to use to build this target.
# At least C++ 11 is required for the AWS SDK for C++.
set(CMAKE_CXX_STANDARD 11)

# Use the MSVC variable to determine if this is a Windows build.
set(WINDOWS_BUILD ${MSVC})

if (WINDOWS_BUILD) # Set the location where CMake can find the installed
  libraries for the AWS SDK.
  string(REPLACE ";" "/aws-cpp-sdk-all;" SYSTEM_MODULE_PATH
    "${CMAKE_SYSTEM_PREFIX_PATH}/aws-cpp-sdk-all")
  list(APPEND CMAKE_PREFIX_PATH ${SYSTEM_MODULE_PATH})
endif ()

# Find the AWS SDK for C++ package.
find_package(AWSSDK REQUIRED COMPONENTS ${SERVICE_COMPONENTS})

if (WINDOWS_BUILD AND AWSSDK_INSTALL_AS_SHARED_LIBS)
  # Copy relevant AWS SDK for C++ libraries into the current binary directory
  for running and debugging.

  # set(BIN_SUB_DIR "/Debug") # If you are building from the command line you
  may need to uncomment this
  # and set the proper subdirectory to the executables' location.

  AWSSDK_CPY_DYN_LIBS(SERVICE_COMPONENTS ""
    ${CMAKE_CURRENT_BINARY_DIR}${BIN_SUB_DIR})
endif ()

add_executable(${PROJECT_NAME}
  hello_sns.cpp)

target_link_libraries(${PROJECT_NAME}
  ${AWSSDK_LINK_LIBRARIES})
```

Código para o arquivo de origem hello\_sns.cpp.

```
#include <aws/core/Aws.h>
```

```
#include <aws/sns/SNSClient.h>
#include <aws/sns/model/ListTopicsRequest.h>
#include <iostream>

/*
 * A "Hello SNS" starter application which initializes an Amazon Simple
 Notification
 * Service (Amazon SNS) client and lists the SNS topics in the current account.
 *
 * main function
 *
 * Usage: 'hello_sns'
 *
 */

int main(int argc, char **argv) {
    Aws::SDKOptions options;
    // Optionally change the log level for debugging.
    // options.loggingOptions.logLevel = Utils::Logging::LogLevel::Debug;
    Aws::InitAPI(options); // Should only be called once.
    {
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";

        Aws::SNS::SNSClient snsClient(clientConfig);

        Aws::Vector<Aws::SNS::Model::Topic> allTopics;
        Aws::String nextToken; // Next token is used to handle a paginated
response.
        do {
            Aws::SNS::Model::ListTopicsRequest request;

            if (!nextToken.empty()) {
                request.SetNextToken(nextToken);
            }

            const Aws::SNS::Model::ListTopicsOutcome outcome =
snsClient.ListTopics(
                request);

            if (outcome.IsSuccess()) {
                const Aws::Vector<Aws::SNS::Model::Topic> &paginatedTopics =
outcome.GetResult().GetTopics();
```

```
        if (!paginatedTopics.empty()) {
            allTopics.insert(allTopics.cend(), paginatedTopics.cbegin(),
                            paginatedTopics.cend());
        }
    }
    else {
        std::cerr << "Error listing topics " <<
outcome.GetError().GetMessage()
        << std::endl;
        return 1;
    }

    nextToken = outcome.GetResult().GetNextToken();
} while (!nextToken.empty());

std::cout << "Hello Amazon SNS! You have " << allTopics.size() << "
topic"
        << (allTopics.size() == 1 ? "" : "s") << " in your account."
        << std::endl;

if (!allTopics.empty()) {
    std::cout << "Here are your topic ARNs." << std::endl;
    for (const Aws::SNS::Model::Topic &topic: allTopics) {
        std::cout << " * " << topic.GetTopicArn() << std::endl;
    }
}

Aws::ShutdownAPI(options); // Should only be called once.
return 0;
}
```

- Para obter detalhes da API, consulte [ListTopics](#) na Referência AWS SDK for C++ da API.

## Go

## SDK para Go V2

 Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
package main

import (
    "context"
    "fmt"
    "log"

    "github.com/aws/aws-sdk-go-v2/config"
    "github.com/aws/aws-sdk-go-v2/service/sns"
    "github.com/aws/aws-sdk-go-v2/service/sns/types"
)

// main uses the AWS SDK for Go V2 to create an Amazon Simple Notification
// Service
// (Amazon SNS) client and list the topics in your account.
// This example uses the default settings specified in your shared credentials
// and config files.
func main() {
    sdkConfig, err := config.LoadDefaultConfig(context.TODO())
    if err != nil {
        fmt.Println("Couldn't load default configuration. Have you set up your AWS
account?")
        fmt.Println(err)
        return
    }
    snsClient := sns.NewFromConfig(sdkConfig)
    fmt.Println("Let's list the topics for your account.")
    var topics []types.Topic
    paginator := sns.NewListTopicsPaginator(snsClient, &sns.ListTopicsInput{})
    for paginator.HasMorePages() {
        output, err := paginator.NextPage(context.TODO())
    }
}
```

```
if err != nil {
    log.Printf("Couldn't get topics. Here's why: %v\n", err)
    break
} else {
    topics = append(topics, output.Topics...)
}
}
if len(topics) == 0 {
    fmt.Println("You don't have any topics!")
} else {
    for _, topic := range topics {
        fmt.Printf("\t%v\n", *topic.TopicArn)
    }
}
}
```

- Para obter detalhes da API, consulte [ListTopics](#) na Referência AWS SDK for Go da API.

## Java

### SDK para Java 2.x

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
package com.example.sns;

import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.sns.SnsClient;
import software.amazon.awssdk.services.sns.model.SnsException;
import software.amazon.awssdk.services.sns.paginators.ListTopicsIterable;

public class HelloSNS {
    public static void main(String[] args) {
        SnsClient snsClient = SnsClient.builder()
            .region(Region.US_EAST_1)
            .build();
```

```
        listSNSTopics(snsClient);
        snsClient.close();
    }

    public static void listSNSTopics(SnsClient snsClient) {
        try {
            ListTopicsIterable listTopics = snsClient.listTopicsPaginator();
            listTopics.stream()
                .flatMap(r -> r.topics().stream())
                .forEach(content -> System.out.println(" Topic ARN: " +
content.topicArn()));

        } catch (SnsException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Para obter detalhes da API, consulte [ListTopics](#) Referência AWS SDK for Java 2.x da API.

## JavaScript

### SDK para JavaScript (v3)

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

Inicialize um cliente SNS e liste tópicos em sua conta.

```
import { SNSClient, paginateListTopics } from "@aws-sdk/client-sns";

export const helloSns = async () => {
    // The configuration object ( `{}` ) is required. If the region and credentials
    // are omitted, the SDK uses your local configuration if it exists.
    const client = new SNSClient({});
```

```
// You can also use `ListTopicsCommand`, but to use that command you must
// handle the pagination yourself. You can do that by sending the
`ListTopicsCommand`
// with the `NextToken` parameter from the previous request.
const paginatedTopics = paginateListTopics({ client }, {});
const topics = [];

for await (const page of paginatedTopics) {
  if (page.Topics?.length) {
    topics.push(...page.Topics);
  }
}

const suffix = topics.length === 1 ? "" : "s";

console.log(
  `Hello, Amazon SNS! You have ${topics.length} topic${suffix} in your
  account.`
);
console.log(topics.map((t) => ` * ${t.TopicArn}`).join("\n"));
};
```

- Para obter detalhes da API, consulte [ListTopics](#) na Referência AWS SDK for JavaScript da API.

## Kotlin

### SDK para Kotlin

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
import aws.sdk.kotlin.services.sns.SnsClient
import aws.sdk.kotlin.services.sns.model.ListTopicsRequest
import aws.sdk.kotlin.services.sns.paginators.listTopicsPaginated
import kotlinx.coroutines.flow.transform
```

```
/**
Before running this Kotlin code example, set up your development environment,
including your credentials.

For more information, see the following documentation topic:
https://docs.aws.amazon.com/sdk-for-kotlin/latest/developer-guide/setup.html
*/
suspend fun main() {
    listTopicsPag()
}

suspend fun listTopicsPag() {
    SnsClient { region = "us-east-1" }.use { snsClient ->
        snsClient.listTopicsPaginated(ListTopicsRequest { })
            .transform { it.topics?.forEach { topic -> emit(topic) } }
            .collect { topic ->
                println("The topic ARN is ${topic.topicArn}")
            }
    }
}
}
```

- Para obter detalhes da API, consulte a [ListTopics](#) referência da API AWS SDK for Kotlin.

## Exemplos de código

- [Ações para o Amazon SNS usando SDKs AWS](#)
  - [Use CheckIfPhoneNumberIsOptedOut com um AWS SDK ou CLI](#)
  - [Use ConfirmSubscription com um AWS SDK ou CLI](#)
  - [Use CreateTopic com um AWS SDK ou CLI](#)
  - [Use DeleteTopic com um AWS SDK ou CLI](#)
  - [Use GetSMSAttributes com um AWS SDK ou CLI](#)
  - [Use GetTopicAttributes com um AWS SDK ou CLI](#)
  - [Use ListPhoneNumbersOptedOut com um AWS SDK ou CLI](#)
  - [Use ListSubscriptions com um AWS SDK ou CLI](#)
  - [Use ListTopics com um AWS SDK ou CLI](#)
  - [Use Publish com um AWS SDK ou CLI](#)
  - [Use SetSMSAttributes com um AWS SDK ou CLI](#)

- [Use SetSubscriptionAttributes com um AWS SDK ou CLI](#)
- [Use SetSubscriptionAttributesRedrivePolicy com um AWS SDK ou CLI](#)
- [Use SetTopicAttributes com um AWS SDK ou CLI](#)
- [Use Subscribe com um AWS SDK ou CLI](#)
- [Use TagResource com um AWS SDK ou CLI](#)
- [Use Unsubscribe com um AWS SDK ou CLI](#)
- [Cenários para o Amazon SNS usando SDKs AWS](#)
  - [Crie um endpoint de plataforma para notificações push do Amazon SNS usando um SDK AWS](#)
  - [Crie e publique em um tópico FIFO do Amazon SNS usando um SDK AWS](#)
  - [Publique mensagens SMS em um tópico do Amazon SNS usando um SDK AWS](#)
  - [Publique uma mensagem grande no Amazon SNS com o Amazon S3 usando um SDK AWS](#)
  - [Publique uma mensagem de texto SMS do Amazon SNS usando um SDK AWS](#)
  - [Publique mensagens do Amazon SNS nas filas do Amazon SQS usando um SDK AWS](#)
- [Exemplos sem servidor para Amazon SNS usando SDKs AWS](#)
  - [Invocar uma função do Lambda em um acionador do Amazon SNS](#)
- [Exemplos de serviços cruzados para o Amazon SNS usando SDKs AWS](#)
  - [Criar uma aplicação para enviar dados para uma tabela do DynamoDB](#)
  - [Criar uma aplicação de publicação e assinatura que traduz mensagens](#)
  - [Criar uma aplicação de gerenciamento de ativos de fotos que permita que os usuários gerenciem fotos usando rótulos](#)
  - [Criar uma aplicação de exploração do Amazon Textract](#)
  - [Detecte pessoas e objetos em um vídeo com o Amazon Rekognition usando um SDK AWS](#)
  - [Publique mensagens do Amazon SNS nas filas do Amazon SQS usando um SDK AWS](#)
  - [Usar o API Gateway para invocar uma função do Lambda](#)
  - [Usar eventos programados para invocar uma função do Lambda](#)

## Ações para o Amazon SNS usando SDKs AWS

Os exemplos de código a seguir demonstram como realizar ações individuais do Amazon SNS com [AWS SDKs](#). Esses trechos chamam a API do Amazon SNS e são trechos de código de programas

maiores que devem ser executados no contexto. Cada exemplo inclui um link para GitHub, onde você pode encontrar instruções para configurar e executar o código.

Os exemplos a seguir incluem apenas as ações mais utilizadas. Para obter uma lista completa, consulte a [Referência de API do Amazon Simple Notification Service \(Amazon SNS\)](#).

## Exemplos

- [Use CheckIfPhoneNumberIsOptedOut com um AWS SDK ou CLI](#)
- [Use ConfirmSubscription com um AWS SDK ou CLI](#)
- [Use CreateTopic com um AWS SDK ou CLI](#)
- [Use DeleteTopic com um AWS SDK ou CLI](#)
- [Use GetSMSAttributes com um AWS SDK ou CLI](#)
- [Use GetTopicAttributes com um AWS SDK ou CLI](#)
- [Use ListPhoneNumbersOptedOut com um AWS SDK ou CLI](#)
- [Use ListSubscriptions com um AWS SDK ou CLI](#)
- [Use ListTopics com um AWS SDK ou CLI](#)
- [Use Publish com um AWS SDK ou CLI](#)
- [Use SetSMSAttributes com um AWS SDK ou CLI](#)
- [Use SetSubscriptionAttributes com um AWS SDK ou CLI](#)
- [Use SetSubscriptionAttributesRedrivePolicy com um AWS SDK ou CLI](#)
- [Use SetTopicAttributes com um AWS SDK ou CLI](#)
- [Use Subscribe com um AWS SDK ou CLI](#)
- [Use TagResource com um AWS SDK ou CLI](#)
- [Use Unsubscribe com um AWS SDK ou CLI](#)

## Use **CheckIfPhoneNumberIsOptedOut** com um AWS SDK ou CLI

Os exemplos de códigos a seguir mostram como usar `CheckIfPhoneNumberIsOptedOut`.

## .NET

### AWS SDK for .NET

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
using System;
using System.Threading.Tasks;
using Amazon.SimpleNotificationService;
using Amazon.SimpleNotificationService.Model;

/// <summary>
/// This example shows how to use the Amazon Simple Notification Service
/// (Amazon SNS) to check whether a phone number has been opted out.
/// </summary>
public class IsPhoneNumOptedOut
{
    public static async Task Main()
    {
        string phoneNumber = "+15551112222";

        IAmazonSimpleNotificationService client = new
AmazonSimpleNotificationServiceClient();

        await CheckIfOptedOutAsync(client, phoneNumber);
    }

    /// <summary>
    /// Checks to see if the supplied phone number has been opted out.
    /// </summary>
    /// <param name="client">The initialized Amazon SNS Client object used
    /// to check if the phone number has been opted out.</param>
    /// <param name="phoneNumber">A string representing the phone number
    /// to check.</param>
    public static async Task
CheckIfOptedOutAsync(IAmazonSimpleNotificationService client, string
phoneNumber)
    {
```

```
var request = new CheckIfPhoneNumberIsOptedOutRequest
{
    PhoneNumber = phoneNumber,
};

try
{
    var response = await
client.CheckIfPhoneNumberIsOptedOutAsync(request);

    if (response.HttpStatusCode == System.Net.HttpStatusCode.OK)
    {
        string optOutStatus = response.IsOptedOut ? "opted out" :
"not opted out.";
        Console.WriteLine($"The phone number: {phoneNumber} is
{optOutStatus}");
    }
    catch (AuthorizationErrorException ex)
    {
        Console.WriteLine($"{ex.Message}");
    }
}
}
```

- Para obter detalhes da API, consulte [CheckIfPhoneNumberIsOptedOut](#) Referência AWS SDK for .NET da API.

## CLI

### AWS CLI

Para verificar o cancelamento de mensagens SMS para um número de telefone

O `check-if-phone-number-is-opted-out` exemplo a seguir verifica se o número de telefone especificado optou por não receber mensagens SMS da AWS conta atual.

```
aws sns check-if-phone-number-is-opted-out \
  --phone-number +1555550100
```

**Saída:**

```
{
  "isOptedOut": false
}
```

- Para obter detalhes da API, consulte [CheckIfPhoneNumberIsOptedOut](#) em Referência de AWS CLI Comandos.

**Java****SDK para Java 2.x****Note**

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.sns.SnsClient;
import
  software.amazon.awssdk.services.sns.model.CheckIfPhoneNumberIsOptedOutRequest;
import
  software.amazon.awssdk.services.sns.model.CheckIfPhoneNumberIsOptedOutResponse;
import software.amazon.awssdk.services.sns.model.SnsException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
 * started.html
 */
public class CheckOptOut {
    public static void main(String[] args) {

        final String usage = ""
```

```
        Usage:    <phoneNumber>

        Where:
            phoneNumber - The mobile phone number to look up (for example,
+1XXX5550100).

        """;

    if (args.length != 1) {
        System.out.println(usage);
        System.exit(1);
    }

    String phoneNumber = args[0];
    SnsClient snsClient = SnsClient.builder()
        .region(Region.US_EAST_1)
        .build();

    checkPhone(snsClient, phoneNumber);
    snsClient.close();
}

public static void checkPhone(SnsClient snsClient, String phoneNumber) {
    try {
        CheckIfPhoneNumberIsOptedOutRequest request =
CheckIfPhoneNumberIsOptedOutRequest.builder()
            .phoneNumber(phoneNumber)
            .build();

        CheckIfPhoneNumberIsOptedOutResponse result =
snsClient.checkIfPhoneNumberIsOptedOut(request);
        System.out.println(
            result.isOptedOut() + "Phone Number " + phoneNumber + " has
Opted Out of receiving sns messages." +
                "\n\nStatus was " +
result.sdkHttpResponse().statusCode());

    } catch (SnsException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}
}
```

- Para obter detalhes da API, consulte [CheckIfPhoneNumberIsOptedOut](#) Referência AWS SDK for Java 2.x da API.

## JavaScript

### SDK para JavaScript (v3)

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

Crie o cliente em um módulo separado e exporte-o.

```
import { SNSClient } from "@aws-sdk/client-sns";

// The AWS Region can be provided here using the `region` property. If you leave
// it blank
// the SDK will default to the region set in your AWS config.
export const snsClient = new SNSClient({});
```

Importe o SDK e os módulos do cliente e chame a API.

```
import { CheckIfPhoneNumberIsOptedOutCommand } from "@aws-sdk/client-sns";

import { snsClient } from "../libs/snsClient.js";

export const checkIfPhoneNumberIsOptedOut = async (
  phoneNumber = "5555555555",
) => {
  const command = new CheckIfPhoneNumberIsOptedOutCommand({
    phoneNumber,
  });

  const response = await snsClient.send(command);
  console.log(response);
}
```

```
// {
//   '$metadata': {
//     httpStatusCode: 200,
//     requestId: '3341c28a-cdc8-5b39-a3ee-9fb0ee125732',
//     extendedRequestId: undefined,
//     cfId: undefined,
//     attempts: 1,
//     totalRetryDelay: 0
//   },
//   isOptedOut: false
// }
return response;
};
```

- Para obter mais informações, consulte o [Guia do desenvolvedor do AWS SDK for JavaScript](#).
- Para obter detalhes da API, consulte [CheckIfPhoneNumberIsOptedOuta Referência AWS SDK for JavaScript da API](#).

## PHP

### SDK para PHP

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
require 'vendor/autoload.php';

use Aws\Exception\AwsException;
use Aws\Sns\SnsClient;

/**
 * Indicates whether the phone number owner has opted out of receiving SMS
 * messages from your AWS SNS account.
 *
 * This code expects that you have AWS credentials set up per:
```

```
* https://docs.aws.amazon.com/sdk-for-php/v3/developer-guide/
guide_credentials.html
*/

$SnSClient = new SnsClient([
    'profile' => 'default',
    'region' => 'us-east-1',
    'version' => '2010-03-31'
]);

$phone = '+1XXX5550100';

try {
    $result = $SnSClient->checkIfPhoneNumberIsOptedOut([
        'phoneNumber' => $phone,
    ]);
    var_dump($result);
} catch (AwsException $e) {
    // output error message if fails
    error_log($e->getMessage());
}
```

- Para obter mais informações, consulte o [Guia do desenvolvedor do AWS SDK for PHP](#).
- Para obter detalhes da API, consulte [CheckIfPhoneNumberIsOptedOut](#) Referência AWS SDK for PHP da API.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulte [Usando o Amazon SNS com um SDK AWS](#). Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

## Use **ConfirmSubscription** com um AWS SDK ou CLI

Os exemplos de códigos a seguir mostram como usar `ConfirmSubscription`.

CLI

AWS CLI

Para confirmar uma assinatura

O comando `confirm-subscription` a seguir conclui o processo de confirmação iniciado quando você se inscreveu em um tópico do SNS chamado `my-topic`. O parâmetro `--token` vem da mensagem de confirmação enviada ao endpoint de notificação especificado na chamada de assinatura.

```
aws sns confirm-subscription \  
  --topic-arn arn:aws:sns:us-west-2:123456789012:my-topic \  
  --token  
2336412f37fb687f5d51e6e241d7700ae02f7124d8268910b858cb4db727ceeb2474bb937929d3bdd7ce5d0c
```

Saída:

```
{  
  "SubscriptionArn": "arn:aws:sns:us-west-2:123456789012:my-  
topic:8a21d249-4329-4871-acc6-7be709c6ea7f"  
}
```

- Para obter detalhes da API, consulte [ConfirmSubscription](#) em Referência de AWS CLI Comandos.

## Java

### SDK para Java 2.x

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
import software.amazon.awssdk.regions.Region;  
import software.amazon.awssdk.services.sns.SnsClient;  
import software.amazon.awssdk.services.sns.model.ConfirmSubscriptionRequest;  
import software.amazon.awssdk.services.sns.model.ConfirmSubscriptionResponse;  
import software.amazon.awssdk.services.sns.model.SnsException;  
  
/**  
 * Before running this Java V2 code example, set up your development  
 * environment, including your credentials.  
 */
```

```
* For more information, see the following documentation topic:
*
* https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
*/
public class ConfirmSubscription {
    public static void main(String[] args) {
        final String usage = ""

            Usage:    <subscriptionToken> <topicArn>

            Where:
                subscriptionToken - A short-lived token sent to an endpoint
                during the Subscribe action.
                topicArn - The ARN of the topic.\s
            """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String subscriptionToken = args[0];
        String topicArn = args[1];
        SnsClient snsClient = SnsClient.builder()
            .region(Region.US_EAST_1)
            .build();

        confirmSub(snsClient, subscriptionToken, topicArn);
        snsClient.close();
    }

    public static void confirmSub(SnsClient snsClient, String subscriptionToken,
        String topicArn) {
        try {
            ConfirmSubscriptionRequest request =
                ConfirmSubscriptionRequest.builder()
                    .token(subscriptionToken)
                    .topicArn(topicArn)
                    .build();

            ConfirmSubscriptionResponse result =
                snsClient.confirmSubscription(request);
        }
    }
}
```

```
        System.out.println("\n\nStatus was " +
result.sdkHttpResponse().statusCode() + "\n\nSubscription Arn: \n\n"
        + result.subscriptionArn());

    } catch (SnsException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}
}
```

- Para obter detalhes da API, consulte [ConfirmSubscription](#) Referência AWS SDK for Java 2.x da API.

## JavaScript

### SDK para JavaScript (v3)

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

Crie o cliente em um módulo separado e exporte-o.

```
import { SNSClient } from "@aws-sdk/client-sns";

// The AWS Region can be provided here using the `region` property. If you leave
it blank
// the SDK will default to the region set in your AWS config.
export const snsClient = new SNSClient({});
```

Importe o SDK e os módulos do cliente e chame a API.

```
import { ConfirmSubscriptionCommand } from "@aws-sdk/client-sns";
import { snsClient } from "../libs/snsClient.js";
```

```
/**
 * @param {string} token - This token is sent the subscriber. Only subscribers
 *                        that are not AWS services (HTTP/S, email) need to be
 *                        confirmed.
 * @param {string} topicArn - The ARN of the topic for which you wish to confirm
 *                            a subscription.
 */
export const confirmSubscription = async (
  token = "TOKEN",
  topicArn = "TOPIC_ARN",
) => {
  const response = await snsClient.send(
    // A subscription only needs to be confirmed if the endpoint type is
    // HTTP/S, email, or in another AWS account.
    new ConfirmSubscriptionCommand({
      Token: token,
      TopicArn: topicArn,
      // If this is true, the subscriber cannot unsubscribe while
      unauthenticated: true,
      AuthenticateOnUnsubscribe: "false",
    }),
  );
  console.log(response);
  // {
  //   '$metadata': {
  //     httpStatusCode: 200,
  //     requestId: '4bb5bce9-805a-5517-8333-e1d2cface90b',
  //     extendedRequestId: undefined,
  //     cfId: undefined,
  //     attempts: 1,
  //     totalRetryDelay: 0
  //   },
  //   SubscriptionArn: 'arn:aws:sns:us-east-1:xxxxxxxxxxxxx:TOPIC_NAME:xxxxxxxx-
  xxx-xxxx-xxxx-xxxxxxxxxxxxx'
  // }
  return response;
};
```

- Para obter mais informações, consulte o [Guia do desenvolvedor do AWS SDK for JavaScript](#).
- Para obter detalhes da API, consulte [ConfirmSubscription](#) na Referência AWS SDK for JavaScript da API.

## PHP

## SDK para PHP

 Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
require 'vendor/autoload.php';

use Aws\Exception\AwsException;
use Aws\Sns\SnsClient;

/**
 * Verifies an endpoint owner's intent to receive messages by
 * validating the token sent to the endpoint by an earlier Subscribe action.
 *
 * This code expects that you have AWS credentials set up per:
 * https://docs.aws.amazon.com/sdk-for-php/v3/developer-guide/
 * guide_credentials.html
 */

$SnSClient = new SnsClient([
    'profile' => 'default',
    'region' => 'us-east-1',
    'version' => '2010-03-31'
]);

$subscription_token = 'arn:aws:sns:us-east-1:111122223333:MyTopic:123456-
abcd-12ab-1234-12ba3dc1234a';
$topic = 'arn:aws:sns:us-east-1:111122223333:MyTopic';

try {
    $result = $SnSClient->confirmSubscription([
        'Token' => $subscription_token,
        'TopicArn' => $topic,
    ]);
    var_dump($result);
} catch (AwsException $e) {
```

```
// output error message if fails
error_log($e->getMessage());
}
```

- Para obter detalhes da API, consulte [ConfirmSubscription](#) na Referência AWS SDK for PHP da API.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulte [Usando o Amazon SNS com um SDK AWS](#). Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

## Use **CreateTopic** com um AWS SDK ou CLI

Os exemplos de códigos a seguir mostram como usar `CreateTopic`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto nos seguintes exemplos de código:

- [Criar e publicar em um tópico FIFO](#)
- [Publicar mensagens em filas](#)

### .NET

#### AWS SDK for .NET

##### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

Crie um tópico com um nome específico.

```
using System;
using System.Threading.Tasks;
using Amazon.SimpleNotificationService;
using Amazon.SimpleNotificationService.Model;
```

```
/// <summary>
/// This example shows how to use Amazon Simple Notification Service
/// (Amazon SNS) to add a new Amazon SNS topic.
/// </summary>
public class CreateSNSTopic
{
    public static async Task Main()
    {
        string topicName = "ExampleSNSTopic";

        IAmazonSimpleNotificationService client = new
AmazonSimpleNotificationServiceClient();

        var topicArn = await CreateSNSTopicAsync(client, topicName);
        Console.WriteLine($"New topic ARN: {topicArn}");
    }

    /// <summary>
    /// Creates a new SNS topic using the supplied topic name.
    /// </summary>
    /// <param name="client">The initialized SNS client object used to
    /// create the new topic.</param>
    /// <param name="topicName">A string representing the topic name.</param>
    /// <returns>The Amazon Resource Name (ARN) of the created topic.</
returns>
    public static async Task<string>
CreateSNSTopicAsync(IAmazonSimpleNotificationService client, string topicName)
    {
        var request = new CreateTopicRequest
        {
            Name = topicName,
        };

        var response = await client.CreateTopicAsync(request);

        return response.TopicArn;
    }
}
```

Crie um tópico com um nome e atributos específicos de FIFO e deduplicação.

```
/// <summary>
/// Create a new topic with a name and specific FIFO and de-duplication
attributes.
/// </summary>
/// <param name="topicName">The name for the topic.</param>
/// <param name="useFifoTopic">True to use a FIFO topic.</param>
/// <param name="useContentBasedDeduplication">True to use content-based de-
duplication.</param>
/// <returns>The ARN of the new topic.</returns>
public async Task<string> CreateTopicWithName(string topicName, bool
useFifoTopic, bool useContentBasedDeduplication)
{
    var createTopicRequest = new CreateTopicRequest()
    {
        Name = topicName,
    };

    if (useFifoTopic)
    {
        // Update the name if it is not correct for a FIFO topic.
        if (!topicName.EndsWith(".fifo"))
        {
            createTopicRequest.Name = topicName + ".fifo";
        }

        // Add the attributes from the method parameters.
        createTopicRequest.Attributes = new Dictionary<string, string>
        {
            { "FifoTopic", "true" }
        };
        if (useContentBasedDeduplication)
        {
            createTopicRequest.Attributes.Add("ContentBasedDeduplication",
"true");
        }
    }

    var createResponse = await
_amazonSNSClient.CreateTopicAsync(createTopicRequest);
    return createResponse.TopicArn;
}
```

- Para obter detalhes da API, consulte [CreateTopic](#) Referência AWS SDK for .NET da API.

## C++

### SDK para C++

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
#!/ Create an Amazon Simple Notification Service (Amazon SNS) topic.
/ *!
  \param topicName: An Amazon SNS topic name.
  \param topicARNResult: String to return the Amazon Resource Name (ARN) for the
  topic.
  \param clientConfiguration: AWS client configuration.
  \return bool: Function succeeded.
 */
bool AwsDoc::SNS::createTopic(const Aws::String &topicName,
                              Aws::String &topicARNResult,
                              const Aws::Client::ClientConfiguration
&clientConfiguration) {
    Aws::SNS::SNSClient snsClient(clientConfiguration);

    Aws::SNS::Model::CreateTopicRequest request;
    request.SetName(topicName);

    const Aws::SNS::Model::CreateTopicOutcome outcome =
snsClient.CreateTopic(request);

    if (outcome.IsSuccess()) {
        topicARNResult = outcome.GetResult().GetTopicArn();
        std::cout << "Successfully created an Amazon SNS topic " << topicName
                  << " with topic ARN '" << topicARNResult
                  << "'." << std::endl;
    }
    else {
        std::cerr << "Error creating topic " << topicName << ":" <<
outcome.GetError().GetMessage() << std::endl;
    }
}
```

```
        topicARNResult.clear();
    }

    return outcome.IsSuccess();
}
```

- Para obter detalhes da API, consulte [CreateTopic](#) Referência AWS SDK for C++ da API.

## CLI

### AWS CLI

Para criar um tópico do SNS

O exemplo `create-topic` a seguir cria um tópico do SNS chamado `my-topic`.

```
aws sns create-topic \
  --name my-topic
```

Saída:

```
{
  "ResponseMetadata": {
    "RequestId": "1469e8d7-1642-564e-b85d-a19b4b341f83"
  },
  "TopicArn": "arn:aws:sns:us-west-2:123456789012:my-topic"
}
```

Para obter mais informações, consulte [Usando a interface de linha de AWS comando com o Amazon SQS e o Amazon SNS](#) no Guia do usuário AWS da interface de linha de comando.

- Para obter detalhes da API, consulte [CreateTopic](#)em Referência de AWS CLI Comandos.

## Go

## SDK para Go V2

 Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
// SnsActions encapsulates the Amazon Simple Notification Service (Amazon SNS)
// actions
// used in the examples.
type SnsActions struct {
    SnsClient *sns.Client
}

// CreateTopic creates an Amazon SNS topic with the specified name. You can
// optionally
// specify that the topic is created as a FIFO topic and whether it uses content-
// based
// deduplication instead of ID-based deduplication.
func (actor SnsActions) CreateTopic(topicName string, isFifoTopic bool,
    contentBasedDeduplication bool) (string, error) {
    var topicArn string
    topicAttributes := map[string]string{}
    if isFifoTopic {
        topicAttributes["FifoTopic"] = "true"
    }
    if contentBasedDeduplication {
        topicAttributes["ContentBasedDeduplication"] = "true"
    }
    topic, err := actor.SnsClient.CreateTopic(context.TODO(), &sns.CreateTopicInput{
        Name:      aws.String(topicName),
        Attributes: topicAttributes,
    })
    if err != nil {
        log.Printf("Couldn't create topic %v. Here's why: %v\n", topicName, err)
    } else {
```

```
    topicArn = *topic.TopicArn
  }

  return topicArn, err
}
```

- Para obter detalhes da API, consulte [CreateTopic](#) Referência AWS SDK for Go da API.

## Java

### SDK para Java 2.x

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.sns.SnsClient;
import software.amazon.awssdk.services.sns.model.CreateTopicRequest;
import software.amazon.awssdk.services.sns.model.CreateTopicResponse;
import software.amazon.awssdk.services.sns.model.SnsException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
 * started.html
 */
public class CreateTopic {
    public static void main(String[] args) {
        final String usage = ""

            Usage:    <topicName>

            Where:
```

```
        topicName - The name of the topic to create (for example,
mytopic).

        """;

    if (args.length != 1) {
        System.out.println(usage);
        System.exit(1);
    }

    String topicName = args[0];
    System.out.println("Creating a topic with name: " + topicName);
    SnsClient snsClient = SnsClient.builder()
        .region(Region.US_EAST_1)
        .build();

    String arnVal = createSNSTopic(snsClient, topicName);
    System.out.println("The topic ARN is" + arnVal);
    snsClient.close();
}

public static String createSNSTopic(SnsClient snsClient, String topicName) {
    CreateTopicResponse result;
    try {
        CreateTopicRequest request = CreateTopicRequest.builder()
            .name(topicName)
            .build();

        result = snsClient.createTopic(request);
        return result.topicArn();

    } catch (SnsException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
    return "";
}
}
```

- Para obter detalhes da API, consulte [CreateTopic](#) Referência AWS SDK for Java 2.x da API.

## JavaScript

### SDK para JavaScript (v3)

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

Crie o cliente em um módulo separado e exporte-o.

```
import { SNSClient } from "@aws-sdk/client-sns";

// The AWS Region can be provided here using the `region` property. If you leave
// it blank
// the SDK will default to the region set in your AWS config.
export const snsClient = new SNSClient({});
```

Importe o SDK e os módulos do cliente e chame a API.

```
import { CreateTopicCommand } from "@aws-sdk/client-sns";
import { snsClient } from "../libs/snsClient.js";

/**
 * @param {string} topicName - The name of the topic to create.
 */
export const createTopic = async (topicName = "TOPIC_NAME") => {
  const response = await snsClient.send(
    new CreateTopicCommand({ Name: topicName }),
  );
  console.log(response);
  // {
  //   '$metadata': {
  //     httpStatusCode: 200,
  //     requestId: '087b8ad2-4593-50c4-a496-d7e90b82cf3e',
  //     extendedRequestId: undefined,
  //     cfId: undefined,
  //     attempts: 1,
  //     totalRetryDelay: 0
  //   },
  // }
```

```
// TopicArn: 'arn:aws:sns:us-east-1:xxxxxxxxxxxxx:TOPIC_NAME'  
// }  
return response;  
};
```

- Para obter mais informações, consulte o [Guia do desenvolvedor do AWS SDK for JavaScript](#).
- Para obter detalhes da API, consulte [CreateTopic](#) Referência AWS SDK for JavaScript da API.

## Kotlin

### SDK para Kotlin

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
suspend fun createSNSTopic(topicName: String): String {  
  
    val request = CreateTopicRequest {  
        name = topicName  
    }  
  
    SnsClient { region = "us-east-1" }.use { snsClient ->  
        val result = snsClient.createTopic(request)  
        return result.topicArn.toString()  
    }  
}
```

- Para obter detalhes da API, consulte a [CreateTopic](#) referência da API AWS SDK for Kotlin.

## PHP

## SDK para PHP

 Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
require 'vendor/autoload.php';

use Aws\Exception\AwsException;
use Aws\Sns\SnsClient;

/**
 * Create a Simple Notification Service topics in your AWS account at the
 * requested region.
 *
 * This code expects that you have AWS credentials set up per:
 * https://docs.aws.amazon.com/sdk-for-php/v3/developer-guide/
 * guide_credentials.html
 */

$SnSClient = new SnsClient([
    'profile' => 'default',
    'region' => 'us-east-1',
    'version' => '2010-03-31'
]);

$topicname = 'myTopic';

try {
    $result = $SnSClient->createTopic([
        'Name' => $topicname,
    ]);
    var_dump($result);
} catch (AwsException $e) {
    // output error message if fails
    error_log($e->getMessage());
}
```

- Para obter mais informações, consulte o [Guia do desenvolvedor do AWS SDK for PHP](#).
- Para obter detalhes da API, consulte [CreateTopic](#) Referência AWS SDK for PHP da API.

## Python

### SDK para Python (Boto3)

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
class SnsWrapper:
    """Encapsulates Amazon SNS topic and subscription functions."""

    def __init__(self, sns_resource):
        """
        :param sns_resource: A Boto3 Amazon SNS resource.
        """
        self.sns_resource = sns_resource

    def create_topic(self, name):
        """
        Creates a notification topic.

        :param name: The name of the topic to create.
        :return: The newly created topic.
        """
        try:
            topic = self.sns_resource.create_topic(Name=name)
            logger.info("Created topic %s with ARN %s.", name, topic.arn)
        except ClientError:
            logger.exception("Couldn't create topic %s.", name)
            raise
        else:
            return topic
```

- Para obter detalhes da API, consulte a [CreateTopic](#) Referência da API AWS SDK for Python (Boto3).

## Ruby

### SDK para Ruby

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
# This class demonstrates how to create an Amazon Simple Notification Service
(SNS) topic.
class SNSTopicCreator
  # Initializes an SNS client.
  #
  # Utilizes the default AWS configuration for region and credentials.
  def initialize
    @sns_client = Aws::SNS::Client.new
  end

  # Attempts to create an SNS topic with the specified name.
  #
  # @param topic_name [String] The name of the SNS topic to create.
  # @return [Boolean] true if the topic was successfully created, false
  otherwise.
  def create_topic(topic_name)
    @sns_client.create_topic(name: topic_name)
    puts "The topic '#{topic_name}' was successfully created."
    true
  rescue Aws::SNS::Errors::ServiceError => e
    # Handles SNS service errors gracefully.
    puts "Error while creating the topic named '#{topic_name}': #{e.message}"
    false
  end
end
```

```
# Example usage:
if $PROGRAM_NAME == __FILE__
  topic_name = "YourTopicName" # Replace with your topic name
  sns_topic_creator = SNSTopicCreator.new

  puts "Creating the topic '#{topic_name}'..."
  unless sns_topic_creator.create_topic(topic_name)
    puts "The topic was not created. Stopping program."
    exit 1
  end
end
```

- Para obter mais informações, consulte o [Guia do desenvolvedor do AWS SDK for Ruby](#).
- Para obter detalhes da API, consulte [CreateTopic](#) Referência AWS SDK for Ruby da API.

## Rust

### SDK para Rust

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
async fn make_topic(client: &Client, topic_name: &str) -> Result<(), Error> {
  let resp = client.create_topic().name(topic_name).send().await?;

  println!(
    "Created topic with ARN: {}",
    resp.topic_arn().unwrap_or_default()
  );

  Ok(())
}
```

- Para obter detalhes da API, consulte a [CreateTopic](#) referência da API AWS SDK for Rust.

## SAP ABAP

### SDK para SAP ABAP

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
TRY.  
    oo_result = lo_sns->createtopic( iv_name = iv_topic_name ). " oo_result  
is returned for testing purposes. "  
    MESSAGE 'SNS topic created' TYPE 'I'.  
    CATCH /aws1/cx_snstopiclimitexc dex.  
        MESSAGE 'Unable to create more topics. You have reached the maximum  
number of topics allowed.' TYPE 'E'.  
ENDTRY.
```

- Para obter detalhes da API, consulte a [CreateTopic](#) referência da API AWS SDK for SAP ABAP.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulte [Usando o Amazon SNS com um SDK AWS](#). Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

## Use **DeleteTopic** com um AWS SDK ou CLI

Os exemplos de códigos a seguir mostram como usar `DeleteTopic`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação no contexto no seguinte exemplo de código:

- [Publicar mensagens em filas](#)

## .NET

### AWS SDK for .NET

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

Exclua um tópico por meio do respectivo ARN.

```
/// <summary>
/// Delete a topic by its topic ARN.
/// </summary>
/// <param name="topicArn">The ARN of the topic.</param>
/// <returns>True if successful.</returns>
public async Task<bool> DeleteTopicByArn(string topicArn)
{
    var deleteResponse = await _amazonSNSClient.DeleteTopicAsync(
        new DeleteTopicRequest()
        {
            TopicArn = topicArn
        });
    return deleteResponse.HttpStatusCode == HttpStatusCode.OK;
}
```

- Para obter detalhes da API, consulte [DeleteTopic](#) Referência AWS SDK for .NET da API.

## C++

### SDK para C++

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
//! Delete an Amazon Simple Notification Service (Amazon SNS) topic.
/*!
  \param topicARN: The Amazon Resource Name (ARN) for an Amazon SNS topic.
  \param clientConfiguration: AWS client configuration.
  \return bool: Function succeeded.
*/
bool AwsDoc::SNS::deleteTopic(const Aws::String &topicARN,
                              const Aws::Client::ClientConfiguration
&clientConfiguration) {
    Aws::SNS::SNSClient snsClient(clientConfiguration);

    Aws::SNS::Model::DeleteTopicRequest request;
    request.SetTopicArn(topicARN);

    const Aws::SNS::Model::DeleteTopicOutcome outcome =
snsClient.DeleteTopic(request);

    if (outcome.IsSuccess()) {
        std::cout << "Successfully deleted the Amazon SNS topic " << topicARN <<
std::endl;
    }
    else {
        std::cerr << "Error deleting topic " << topicARN << ":" <<
outcome.GetError().GetMessage() << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Para obter detalhes da API, consulte [DeleteTopic](#) Referência AWS SDK for C++ da API.

## CLI

### AWS CLI

Para excluir um tópico do SNS

O exemplo `delete-topic` a seguir exclui o tópico do SNS especificado.

```
aws sns delete-topic \
  --topic-arn "arn:aws:sns:us-west-2:123456789012:my-topic"
```

Este comando não produz saída.

- Para obter detalhes da API, consulte [DeleteTopic](#) em Referência de AWS CLI Comandos.

Go

SDK para Go V2

 Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
// SnsActions encapsulates the Amazon Simple Notification Service (Amazon SNS)
// actions
// used in the examples.
type SnsActions struct {
    SnsClient *sns.Client
}

// DeleteTopic delete an Amazon SNS topic.
func (actor SnsActions) DeleteTopic(topicArn string) error {
    _, err := actor.SnsClient.DeleteTopic(context.TODO(), &sns.DeleteTopicInput{
        TopicArn: aws.String(topicArn)})
    if err != nil {
        log.Printf("Couldn't delete topic %v. Here's why: %v\n", topicArn, err)
    }
    return err
}
```

- Para obter detalhes da API, consulte [DeleteTopic](#) em Referência AWS SDK for Go da API.

## Java

## SDK para Java 2.x

 Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.sns.SnsClient;
import software.amazon.awssdk.services.sns.model.DeleteTopicRequest;
import software.amazon.awssdk.services.sns.model.DeleteTopicResponse;
import software.amazon.awssdk.services.sns.model.SnsException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
 * started.html
 */
public class DeleteTopic {
    public static void main(String[] args) {
        final String usage = ""

                Usage:      <topicArn>

                Where:
                    topicArn - The ARN of the topic to delete.
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String topicArn = args[0];
        SnsClient snsClient = SnsClient.builder()
```

```
        .region(Region.US_EAST_1)
        .build();

        System.out.println("Deleting a topic with name: " + topicArn);
        deleteSNSTopic(snsClient, topicArn);
        snsClient.close();
    }

    public static void deleteSNSTopic(SnsClient snsClient, String topicArn) {
        try {
            DeleteTopicRequest request = DeleteTopicRequest.builder()
                .topicArn(topicArn)
                .build();

            DeleteTopicResponse result = snsClient.deleteTopic(request);
            System.out.println("\n\nStatus was " +
                result.sdkHttpResponse().statusCode());

        } catch (SnsException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Para obter detalhes da API, consulte [DeleteTopica](#) Referência AWS SDK for Java 2.x da API.

## JavaScript

### SDK para JavaScript (v3)

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

Crie o cliente em um módulo separado e exporte-o.

```
import { SNSClient } from "@aws-sdk/client-sns";
```

```
// The AWS Region can be provided here using the `region` property. If you leave
it blank
// the SDK will default to the region set in your AWS config.
export const snsClient = new SNSClient({});
```

Importe o SDK e os módulos do cliente e chame a API.

```
import { DeleteTopicCommand } from "@aws-sdk/client-sns";
import { snsClient } from "../libs/snsClient.js";

/**
 * @param {string} topicArn - The ARN of the topic to delete.
 */
export const deleteTopic = async (topicArn = "TOPIC_ARN") => {
  const response = await snsClient.send(
    new DeleteTopicCommand({ TopicArn: topicArn }),
  );
  console.log(response);
  // {
  //   '$metadata': {
  //     httpStatusCode: 200,
  //     requestId: 'a10e2886-5a8f-5114-af36-75bd39498332',
  //     extendedRequestId: undefined,
  //     cfId: undefined,
  //     attempts: 1,
  //     totalRetryDelay: 0
  //   }
  // }
};
```

- Para obter mais informações, consulte o [Guia do desenvolvedor do AWS SDK for JavaScript](#).
- Para obter detalhes da API, consulte [DeleteTopic](#) Referência AWS SDK for JavaScript da API.

## Kotlin

### SDK para Kotlin

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
suspend fun deleteSNSTopic(topicArnVal: String) {  
  
    val request = DeleteTopicRequest {  
        topicArn = topicArnVal  
    }  
  
    SnsClient { region = "us-east-1" }.use { snsClient ->  
        snsClient.deleteTopic(request)  
        println("$topicArnVal was successfully deleted.")  
    }  
}
```

- Para obter detalhes da API, consulte a [DeleteTopic](#) referência da API AWS SDK for Kotlin.

## PHP

### SDK para PHP

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
require 'vendor/autoload.php';  
  
use Aws\Exception\AwsException;  
use Aws\Sns\SnsClient;
```

```
/**
 * Deletes an SNS topic and all its subscriptions.
 *
 * This code expects that you have AWS credentials set up per:
 * https://docs.aws.amazon.com/sdk-for-php/v3/developer-guide/
guide_credentials.html
 */

$SnSClient = new SnsClient([
    'profile' => 'default',
    'region' => 'us-east-1',
    'version' => '2010-03-31'
]);

$topic = 'arn:aws:sns:us-east-1:111122223333:MyTopic';

try {
    $result = $SnSClient->deleteTopic([
        'TopicArn' => $topic,
    ]);
    var_dump($result);
} catch (AwsException $e) {
    // output error message if fails
    error_log($e->getMessage());
}
```

- Para obter detalhes da API, consulte [DeleteTopic](#) Referência AWS SDK for PHP da API.

## Python

### SDK para Python (Boto3)

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
class SnsWrapper:
```

```
"""Encapsulates Amazon SNS topic and subscription functions."""

def __init__(self, sns_resource):
    """
    :param sns_resource: A Boto3 Amazon SNS resource.
    """
    self.sns_resource = sns_resource

    @staticmethod
    def delete_topic(topic):
        """
        Deletes a topic. All subscriptions to the topic are also deleted.
        """
        try:
            topic.delete()
            logger.info("Deleted topic %s.", topic.arn)
        except ClientError:
            logger.exception("Couldn't delete topic %s.", topic.arn)
            raise
```

- Para obter detalhes da API, consulte a [DeleteTopic](#) Referência da API AWS SDK for Python (Boto3).

## SAP ABAP

### SDK para SAP ABAP

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
TRY.
  lo_sns->deletetopic( iv_topicarn = iv_topic_arn ).
  MESSAGE 'SNS topic deleted.' TYPE 'I'.
CATCH /aws1/cx_snsnotfoundexception.
  MESSAGE 'Topic does not exist.' TYPE 'E'.
```

```
ENDTRY.
```

- Para obter detalhes da API, consulte a [DeleteTopic](#) referência da API AWS SDK for SAP ABAP.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulte [Usando o Amazon SNS com um SDK AWS](#). Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

## Use `GetSMSAttributes` com um AWS SDK ou CLI

Os exemplos de códigos a seguir mostram como usar `GetSMSAttributes`.

C++

SDK para C++

### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
//! Retrieve the default settings for sending SMS messages from your AWS account
  by using
  //! Amazon Simple Notification Service (Amazon SNS).
  /*!
   \param clientConfiguration: AWS client configuration.
   \return bool: Function succeeded.
  */
bool
AwsDoc::SNS::getSMSType(const Aws::Client::ClientConfiguration
&clientConfiguration) {
    Aws::SNS::SNSClient snsClient(clientConfiguration);

    Aws::SNS::Model::GetSMSAttributesRequest request;
    //Set the request to only retrieve the DefaultSMSType setting.
    //Without the following line, GetSMSAttributes would retrieve all settings.
    request.AddAttributes("DefaultSMSType");
```

```
const Aws::SNS::Model::GetSMSAttributesOutcome outcome =
snsClient.GetSMSAttributes(
    request);

if (outcome.IsSuccess()) {
    const Aws::Map<Aws::String, Aws::String> attributes =
        outcome.GetResult().GetAttributes();
    if (!attributes.empty()) {
        for (auto const &att: attributes) {
            std::cout << att.first << ": " << att.second << std::endl;
        }
    }
    else {
        std::cout
            << "AwsDoc::SNS::getSMSType - an empty map of attributes was
retrieved."
            << std::endl;
    }
}
else {
    std::cerr << "Error while getting SMS Type: '"
        << outcome.GetError().GetMessage()
        << "'" << std::endl;
}

return outcome.IsSuccess();
}
```

- Para obter detalhes da API, consulte [GetSMSAttributes](#) na Referência da API AWS SDK for C++ .

## CLI

### AWS CLI

Para listar os atributos padrão da mensagem SMS

O exemplo `get-sms-attributes` a seguir lista os atributos padrão para o envio de mensagens SMS.

```
aws sns get-sms-attributes
```

**Saída:**

```
{
  "attributes": {
    "DefaultSenderId": "MyName"
  }
}
```

- Para obter detalhes da API, consulte [GetSMSAttributes](#) na Referência de comandos da AWS CLI .

**Java****SDK para Java 2.x****Note**

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.sns.SnsClient;
import
  software.amazon.awssdk.services.sns.model.GetSubscriptionAttributesRequest;
import
  software.amazon.awssdk.services.sns.model.GetSubscriptionAttributesResponse;
import software.amazon.awssdk.services.sns.model.SnsException;
import java.util.Iterator;
import java.util.Map;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
 * started.html
 */
public class GetSMSAttributes {
```

```
public static void main(String[] args) {
    final String usage = ""

        Usage:    <topicArn>

        Where:
            topicArn - The ARN of the topic from which to retrieve
attributes.
        """;

    if (args.length != 1) {
        System.out.println(usage);
        System.exit(1);
    }

    String topicArn = args[0];
    SnsClient snsClient = SnsClient.builder()
        .region(Region.US_EAST_1)
        .build();

    getSNSAttrutes(snsClient, topicArn);
    snsClient.close();
}

public static void getSNSAttrutes(SnsClient snsClient, String topicArn) {
    try {
        GetSubscriptionAttributesRequest request =
GetSubscriptionAttributesRequest.builder()
            .subscriptionArn(topicArn)
            .build();

        // Get the Subscription attributes
        GetSubscriptionAttributesResponse res =
snsClient.getSubscriptionAttributes(request);
        Map<String, String> map = res.attributes();

        // Iterate through the map
        Iterator iter = map.entrySet().iterator();
        while (iter.hasNext()) {
            Map.Entry entry = (Map.Entry) iter.next();
            System.out.println("[Key] : " + entry.getKey() + " [Value] : " +
entry.getValue());
        }
    }
}
```

```
    } catch (SnsException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }

    System.out.println("\n\nStatus was good");
}
}
```

- Para obter detalhes da API, consulte [GetSMSAttributes](#) na Referência da API AWS SDK for Java 2.x .

## JavaScript

### SDK para JavaScript (v3)

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

Crie o cliente em um módulo separado e exporte-o.

```
import { SNSClient } from "@aws-sdk/client-sns";

// The AWS Region can be provided here using the `region` property. If you leave
// it blank
// the SDK will default to the region set in your AWS config.
export const snsClient = new SNSClient({});
```

Importe o SDK e os módulos do cliente e chame a API.

```
import { GetSMSAttributesCommand } from "@aws-sdk/client-sns";
import { snsClient } from "../libs/snsClient.js";

export const getSmsAttributes = async () => {
    const response = await snsClient.send(
```

```
// If you have not modified the account-level mobile settings of SNS,  
// the DefaultSMSType is undefined. For this example, it was set to  
// Transactional.  
new GetSMSAttributesCommand({ attributes: ["DefaultSMSType"] } ),  
);  
  
console.log(response);  
// {  
//   '$metadata': {  
//     httpStatusCode: 200,  
//     requestId: '67ad8386-4169-58f1-bdb9-debd281d48d5',  
//     extendedRequestId: undefined,  
//     cfId: undefined,  
//     attempts: 1,  
//     totalRetryDelay: 0  
//   },  
//   attributes: { DefaultSMSType: 'Transactional' }  
// }  
return response;  
};
```

- Para obter mais informações, consulte o [Guia do desenvolvedor do AWS SDK for JavaScript](#).
- Para obter detalhes da API, consulte [GetSMSAttributes](#) na Referência da API AWS SDK for JavaScript .

## PHP

### SDK para PHP

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
require 'vendor/autoload.php';  
  
use Aws\Exception\AwsException;  
use Aws\Sns\SnsClient;
```

```
/**
 * Get the type of SMS Message sent by default from the AWS SNS service.
 *
 * This code expects that you have AWS credentials set up per:
 * https://docs.aws.amazon.com/sdk-for-php/v3/developer-guide/
guide_credentials.html
 */

$SnSClient = new SnsClient([
    'profile' => 'default',
    'region' => 'us-east-1',
    'version' => '2010-03-31'
]);

try {
    $result = $SnSClient->getSMSAttributes([
        'attributes' => ['DefaultSMSType'],
    ]);
    var_dump($result);
} catch (AwsException $e) {
    // output error message if fails
    error_log($e->getMessage());
}
```

- Para obter mais informações, consulte o [Guia do desenvolvedor do AWS SDK for PHP](#).
- Para obter detalhes da API, consulte [GetSMSAttributes](#) na Referência da API AWS SDK for PHP .

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulte [Usando o Amazon SNS com um SDK AWS](#). Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

## Use **GetTopicAttributes** com um AWS SDK ou CLI

Os exemplos de códigos a seguir mostram como usar `GetTopicAttributes`.

## .NET

### AWS SDK for .NET

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
using System;
using System.Collections.Generic;
using System.Threading.Tasks;
using Amazon.SimpleNotificationService;

/// <summary>
/// This example shows how to retrieve the attributes of an Amazon Simple
/// Notification Service (Amazon SNS) topic.
/// </summary>
public class GetTopicAttributes
{
    public static async Task Main()
    {
        string topicArn = "arn:aws:sns:us-
west-2:000000000000:ExampleSNSTopic";
        IAmazonSimpleNotificationService client = new
AmazonSimpleNotificationServiceClient();

        var attributes = await GetTopicAttributesAsync(client, topicArn);
        DisplayTopicAttributes(attributes);
    }

    /// <summary>
    /// Given the ARN of the Amazon SNS topic, this method retrieves the
    topic
    /// attributes.
    /// </summary>
    /// <param name="client">The initialized Amazon SNS client object used
    /// to retrieve the attributes for the Amazon SNS topic.</param>
    /// <param name="topicArn">The ARN of the topic for which to retrieve
    /// the attributes.</param>
    /// <returns>A Dictionary of topic attributes.</returns>
}
```

```
public static async Task<Dictionary<string, string>>
GetTopicAttributesAsync(
    IAmazonSimpleNotificationService client,
    string topicArn)
{
    var response = await client.GetTopicAttributesAsync(topicArn);

    return response.Attributes;
}

/// <summary>
/// This method displays the attributes for an Amazon SNS topic.
/// </summary>
/// <param name="topicAttributes">A Dictionary containing the
/// attributes for an Amazon SNS topic.</param>
public static void DisplayTopicAttributes(Dictionary<string, string>
topicAttributes)
{
    foreach (KeyValuePair<string, string> entry in topicAttributes)
    {
        Console.WriteLine($"{entry.Key}: {entry.Value}\n");
    }
}
}
```

- Para obter detalhes da API, consulte [GetTopicAttributes](#) a Referência AWS SDK for .NET da API.

## C++

### SDK para C++

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
//! Retrieve the properties of an Amazon Simple Notification Service (Amazon SNS)
topic.
/*!
 \param topicARN: The Amazon Resource Name (ARN) for an Amazon SNS topic.
 \param clientConfiguration: AWS client configuration.
 \return bool: Function succeeded.
 */
bool AwsDoc::SNS::getTopicAttributes(const Aws::String &topicARN,
                                     const Aws::Client::ClientConfiguration
&clientConfiguration) {
    Aws::SNS::SNSClient snsClient(clientConfiguration);
    Aws::SNS::Model::GetTopicAttributesRequest request;
    request.SetTopicArn(topicARN);

    const Aws::SNS::Model::GetTopicAttributesOutcome outcome =
snsClient.GetTopicAttributes(
    request);

    if (outcome.IsSuccess()) {
        std::cout << "Topic Attributes:" << std::endl;
        for (auto const &attribute: outcome.GetResult().GetAttributes()) {
            std::cout << " * " << attribute.first << " : " << attribute.second
                << std::endl;
        }
    }
    else {
        std::cerr << "Error while getting Topic attributes "
            << outcome.GetError().GetMessage()
            << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Para obter detalhes da API, consulte [GetTopicAttributes](#) a Referência AWS SDK for C++ da API.

## CLI

### AWS CLI

Para recuperar os atributos de um tópico

O exemplo `get-topic-attributes` a seguir exibe os atributos do tópico especificado.

```
aws sns get-topic-attributes \  
  --topic-arn "arn:aws:sns:us-west-2:123456789012:my-topic"
```

Saída:

```
{  
  "Attributes": {  
    "SubscriptionsConfirmed": "1",  
    "DisplayName": "my-topic",  
    "SubscriptionsDeleted": "0",  
    "EffectiveDeliveryPolicy": "{\"http\":{\"defaultHealthyRetryPolicy\  
\":{\"minDelayTarget\":20,\"maxDelayTarget\":20,\"numRetries\":3,  
\"numMaxDelayRetries\":0,\"numNoDelayRetries\":0,\"numMinDelayRetries\":0,  
\"backoffFunction\":\"linear\"},\"disableSubscriptionOverrides\":false}}\",  
    "Owner": "123456789012",  
    "Policy": "{\"Version\":\"2008-10-17\",\"Id\":\"__default_policy_ID\  
\", \"Statement\": [{\"Sid\":\"__default_statement_ID\", \"Effect\":  
\"Allow\", \"Principal\": {\"AWS\": \"*\"}, \"Action\": [\"SNS:Subscribe\",  
\"SNS:ListSubscriptionsByTopic\", \"SNS:DeleteTopic\", \"SNS:GetTopicAttributes\  
\", \"SNS:Publish\", \"SNS:RemovePermission\", \"SNS:AddPermission\",  
\"SNS:SetTopicAttributes\"], \"Resource\": \"arn:aws:sns:us-west-2:123456789012:my-  
topic\", \"Condition\": {\"StringEquals\": {\"AWS:SourceOwner\":  
\"0123456789012\"}}}]}",  
    "TopicArn": "arn:aws:sns:us-west-2:123456789012:my-topic",  
    "SubscriptionsPending": "0"  
  }  
}
```

- Para obter detalhes da API, consulte [GetTopicAttributes](#) em Referência de AWS CLI Comandos.

## Java

## SDK para Java 2.x

 Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.sns.SnsClient;
import software.amazon.awssdk.services.sns.model.GetTopicAttributesRequest;
import software.amazon.awssdk.services.sns.model.GetTopicAttributesResponse;
import software.amazon.awssdk.services.sns.model.SnsException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
 * started.html
 */
public class GetTopicAttributes {
    public static void main(String[] args) {
        final String usage = ""

                Usage:    <topicArn>

                Where:
                    topicArn - The ARN of the topic to look up.
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String topicArn = args[0];
        SnsClient snsClient = SnsClient.builder()
```

```
        .region(Region.US_EAST_1)
        .build();

        System.out.println("Getting attributes for a topic with name: " +
topicArn);
        getSNSTopicAttributes(snsClient, topicArn);
        snsClient.close();
    }

    public static void getSNSTopicAttributes(SnsClient snsClient, String
topicArn) {
        try {
            GetTopicAttributesRequest request =
GetTopicAttributesRequest.builder()
                .topicArn(topicArn)
                .build();

            GetTopicAttributesResponse result =
snsClient.getTopicAttributes(request);
            System.out.println("\n\nStatus is " +
result.sdkHttpResponse().statusCode() + "\n\nAttributes: \n\n"
                + result.attributes());

        } catch (SnsException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Para obter detalhes da API, consulte [GetTopicAttributes](#) a Referência AWS SDK for Java 2.x da API.

## JavaScript

### SDK para JavaScript (v3)

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

Crie o cliente em um módulo separado e exporte-o.

```
import { SNSClient } from "@aws-sdk/client-sns";

// The AWS Region can be provided here using the `region` property. If you leave
// it blank
// the SDK will default to the region set in your AWS config.
export const snsClient = new SNSClient({});
```

Importe o SDK e os módulos do cliente e chame a API.

```
import { GetTopicAttributesCommand } from "@aws-sdk/client-sns";
import { snsClient } from "../libs/snsClient.js";

/**
 * @param {string} topicArn - The ARN of the topic to retrieve attributes for.
 */
export const getTopicAttributes = async (topicArn = "TOPIC_ARN") => {
  const response = await snsClient.send(
    new GetTopicAttributesCommand({
      TopicArn: topicArn,
    }),
  );
  console.log(response);
  // {
  //   '$metadata': {
  //     httpStatusCode: 200,
  //     requestId: '36b6a24e-5473-5d4e-ac32-ff72d9a73d94',
  //     extendedRequestId: undefined,
  //     cfId: undefined,
  //     attempts: 1,
```

```

//   totalRetryDelay: 0
// },
//   Attributes: {
//     Policy: '{...}',
//     Owner: 'xxxxxxxxxxxxx',
//     SubscriptionsPending: '1',
//     TopicArn: 'arn:aws:sns:us-east-1:xxxxxxxxxxxxx:mytopic',
//     TracingConfig: 'PassThrough',
//     EffectiveDeliveryPolicy: '{"http":{"defaultHealthyRetryPolicy":
{"minDelayTarget":20,"maxDelayTarget":20,"numRetries":3,"numMaxDelayRetries":0,"numNoDelays":0,"numUnhealthyRetries":3,"backoffTimeMillis":100,"maxBackoffTimeMillis":10000,"numUnhealthyRetriesBeforeBackoff":3,"retryPolicyType":"Default"},"headerContentType":"text/plain; charset=UTF-8"}}}',
//     SubscriptionsConfirmed: '0',
//     DisplayName: '',
//     SubscriptionsDeleted: '1'
//   }
// }
return response;
};

```

- Para obter mais informações, consulte o [Guia do desenvolvedor do AWS SDK for JavaScript](#).
- Para obter detalhes da API, consulte [GetTopicAttributes](#) a Referência AWS SDK for JavaScript da API.

SDK para JavaScript (v2)

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

Importe o SDK e os módulos do cliente e chame a API.

```

// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set region
AWS.config.update({ region: "REGION" });

// Create promise and SNS service object
var getTopicAttribsPromise = new AWS.SNS({ apiVersion: "2010-03-31" })

```

```
.getTopicAttributes({ TopicArn: "TOPIC_ARN" })
.promise();

// Handle promise's fulfilled/rejected states
getTopicAttribsPromise
  .then(function (data) {
    console.log(data);
  })
  .catch(function (err) {
    console.error(err, err.stack);
  });
```

- Para obter mais informações, consulte o [Guia do desenvolvedor do AWS SDK for JavaScript](#).
- Para obter detalhes da API, consulte [GetTopicAttributes](#) a Referência AWS SDK for JavaScript da API.

## Kotlin

### SDK para Kotlin

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
suspend fun getSnsTopicAttributes(topicArnVal: String) {

    val request = GetTopicAttributesRequest {
        topicArn = topicArnVal
    }

    SnsClient { region = "us-east-1" }.use { snsClient ->
        val result = snsClient.getTopicAttributes(request)
        println("${result.attributes}")
    }
}
```

- Para obter detalhes da API, consulte a [GetTopicAttributes](#) referência da API AWS SDK for Kotlin.

## PHP

### SDK para PHP

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
$SnSClient = new SnsClient([
    'profile' => 'default',
    'region' => 'us-east-1',
    'version' => '2010-03-31'
]);

$topic = 'arn:aws:sns:us-east-1:111122223333:MyTopic';

try {
    $result = $SnSClient->getTopicAttributes([
        'TopicArn' => $topic,
    ]);
    var_dump($result);
} catch (AwsException $e) {
    // output error message if fails
    error_log($e->getMessage());
}
```

- Para obter detalhes da API, consulte [GetTopicAttributes](#) a Referência AWS SDK for PHP da API.

## SAP ABAP

### SDK para SAP ABAP

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
TRY.  
    oo_result = lo_sns->gettopicattributes( iv_topicarn = iv_topic_arn ). "  
oo_result is returned for testing purposes. "  
    DATA(lt_attributes) = oo_result->get_attributes( ).  
    MESSAGE 'Retrieved attributes/properties of a topic.' TYPE 'I'.  
CATCH /aws1/cx_snsnotfoundexception.  
    MESSAGE 'Topic does not exist.' TYPE 'E'.  
ENDTRY.
```

- Para obter detalhes da API, consulte a [GetTopicAttributes](#) referência da API AWS SDK for SAP ABAP.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulte [Usando o Amazon SNS com um SDK AWS](#). Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

## Use **ListPhoneNumbersOptedOut** com um AWS SDK ou CLI

Os exemplos de códigos a seguir mostram como usar `ListPhoneNumbersOptedOut`.

### CLI

#### AWS CLI

Para listar as opções de cancelamento de mensagens SMS

O exemplo `list-phone-numbers-opted-out` a seguir lista os números de telefone que optaram por não receber mensagens SMS.

```
aws sns list-phone-numbers-opted-out
```

Saída:

```
{
  "phoneNumbers": [
    "+15555550100"
  ]
}
```

- Para obter detalhes da API, consulte [ListPhoneNumbersOptedOut](#) em Referência de AWS CLI Comandos.

Java

SDK para Java 2.x

 Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.sns.SnsClient;
import software.amazon.awssdk.services.sns.model.ListPhoneNumbersOptedOutRequest;
import
  software.amazon.awssdk.services.sns.model.ListPhoneNumbersOptedOutResponse;
import software.amazon.awssdk.services.sns.model.SnsException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
 * started.html
 */
public class ListOptOut {
```

```
public static void main(String[] args) {
    SnsClient snsClient = SnsClient.builder()
        .region(Region.US_EAST_1)
        .build();

    listOpts(snsClient);
    snsClient.close();
}

public static void listOpts(SnsClient snsClient) {
    try {
        ListPhoneNumbersOptedOutRequest request =
ListPhoneNumbersOptedOutRequest.builder().build();
        ListPhoneNumbersOptedOutResponse result =
snsClient.listPhoneNumbersOptedOut(request);
        System.out.println("Status is " +
result.sdkHttpResponse().statusCode() + "\n\nPhone Numbers: \n\n"
            + result.phoneNumbers());

    } catch (SnsException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}
}
```

- Para obter detalhes da API, consulte [ListPhoneNumbersOptedOut](#) Referência AWS SDK for Java 2.x da API.

## PHP

### SDK para PHP

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
require 'vendor/autoload.php';
```

```
use Aws\Exception\AwsException;
use Aws\Sns\SnsClient;

/**
 * Returns a list of phone numbers that are opted out of receiving SMS messages
 * from your AWS SNS account.
 *
 * This code expects that you have AWS credentials set up per:
 * https://docs.aws.amazon.com/sdk-for-php/v3/developer-guide/
 * guide_credentials.html
 */

$SnSClient = new SnsClient([
    'profile' => 'default',
    'region' => 'us-east-1',
    'version' => '2010-03-31'
]);

try {
    $result = $SnSClient->listPhoneNumbersOptedOut();
    var_dump($result);
} catch (AwsException $e) {
    // output error message if fails
    error_log($e->getMessage());
}
```

- Para obter mais informações, consulte o [Guia do desenvolvedor do AWS SDK for PHP](#).
- Para obter detalhes da API, consulte [ListPhoneNumbersOptedOut](#) Referência AWS SDK for PHP da API.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulte [Usando o Amazon SNS com um SDK AWS](#). Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

## Use **ListSubscriptions** com um AWS SDK ou CLI

Os exemplos de códigos a seguir mostram como usar `ListSubscriptions`.

## .NET

### AWS SDK for .NET

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
using System;
using System.Collections.Generic;
using System.Threading.Tasks;
using Amazon.SimpleNotificationService;
using Amazon.SimpleNotificationService.Model;

/// <summary>
/// This example will retrieve a list of the existing Amazon Simple
/// Notification Service (Amazon SNS) subscriptions.
/// </summary>
public class ListSubscriptions
{
    public static async Task Main()
    {
        IAmazonSimpleNotificationService client = new
AmazonSimpleNotificationServiceClient();

        Console.WriteLine("Enter a topic ARN to list subscriptions for a
specific topic, " +
                        "or press Enter to list subscriptions for all
topics.");
        var topicArn = Console.ReadLine();
        Console.WriteLine();

        var subscriptions = await GetSubscriptionsListAsync(client,
topicArn);

        DisplaySubscriptionList(subscriptions);
    }

    /// <summary>
```

```
    /// Gets a list of the existing Amazon SNS subscriptions, optionally by
    /// specifying a topic ARN.
    /// </summary>
    /// <param name="client">The initialized Amazon SNS client object used
    /// to obtain the list of subscriptions.</param>
    /// <param name="topicArn">The optional ARN of a specific topic. Defaults
    /// to null.</param>
    /// <returns>A list containing information about each subscription.</
returns>
    public static async Task<List<Subscription>>
GetSubscriptionsListAsync(IAmazonSimpleNotificationService client, string
topicArn = null)
    {
        var results = new List<Subscription>();

        if (!string.IsNullOrEmpty(topicArn))
        {
            var paginateByTopic = client.Paginators.ListSubscriptionsByTopic(
                new ListSubscriptionsByTopicRequest()
                {
                    TopicArn = topicArn,
                });

            // Get the entire list using the paginator.
            await foreach (var subscription in paginateByTopic.Subscriptions)
            {
                results.Add(subscription);
            }
        }
        else
        {
            var paginateAllSubscriptions =
client.Paginators.ListSubscriptions(new ListSubscriptionsRequest());

            // Get the entire list using the paginator.
            await foreach (var subscription in
paginateAllSubscriptions.Subscriptions)
            {
                results.Add(subscription);
            }
        }

        return results;
    }
}
```

```
/// <summary>
/// Display a list of Amazon SNS subscription information.
/// </summary>
/// <param name="subscriptionList">A list containing details for existing
/// Amazon SNS subscriptions.</param>
public static void DisplaySubscriptionList(List<Subscription>
subscriptionList)
{
    foreach (var subscription in subscriptionList)
    {
        Console.WriteLine($"Owner: {subscription.Owner}");
        Console.WriteLine($"Subscription ARN:
{subscription.SubscriptionArn}");
        Console.WriteLine($"Topic ARN: {subscription.TopicArn}");
        Console.WriteLine($"Endpoint: {subscription.Endpoint}");
        Console.WriteLine($"Protocol: {subscription.Protocol}");
        Console.WriteLine();
    }
}
}
```

- Para obter detalhes da API, consulte [ListSubscriptions](#) na Referência AWS SDK for .NET da API.

## C++

### SDK para C++

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
//! Retrieve a list of Amazon Simple Notification Service (Amazon SNS)
subscriptions.
/*!
    \param clientConfiguration: AWS client configuration.
```

```
\return bool: Function succeeded.
*/
bool AwsDoc::SNS::listSubscriptions(
    const Aws::Client::ClientConfiguration &clientConfiguration) {
    Aws::SNS::SNSClient snsClient(clientConfiguration);

    Aws::String nextToken; // Next token is used to handle a paginated response.
    bool result = true;
    Aws::Vector<Aws::SNS::Model::Subscription> subscriptions;
    do {
        Aws::SNS::Model::ListSubscriptionsRequest request;

        if (!nextToken.empty()) {
            request.SetNextToken(nextToken);
        }

        const Aws::SNS::Model::ListSubscriptionsOutcome outcome =
snsClient.ListSubscriptions(
            request);

        if (outcome.IsSuccess()) {
            const Aws::Vector<Aws::SNS::Model::Subscription> &newSubscriptions =
                outcome.GetResult().GetSubscriptions();
            subscriptions.insert(subscriptions.cend(), newSubscriptions.begin(),
                newSubscriptions.end());
        }
        else {
            std::cerr << "Error listing subscriptions "
                << outcome.GetError().GetMessage()
                <<
                std::endl;
            result = false;
            break;
        }

        nextToken = outcome.GetResult().GetNextToken();
    } while (!nextToken.empty());

    if (result) {
        if (subscriptions.empty()) {
            std::cout << "No subscriptions found" << std::endl;
        }
        else {
            std::cout << "Subscriptions list:" << std::endl;
        }
    }
}
```

```
        for (auto const &subscription: subscriptions) {
            std::cout << " * " << subscription.GetSubscriptionArn() <<
std::endl;
        }
    }
    return result;
}
```

- Para obter detalhes da API, consulte [ListSubscriptions](#) a Referência AWS SDK for C++ da API.

## CLI

### AWS CLI

Para listar suas assinaturas do SNS

O `list-subscriptions` exemplo a seguir exibe uma lista das assinaturas do SNS em sua conta. AWS

```
aws sns list-subscriptions
```

Saída:

```
{
  "Subscriptions": [
    {
      "Owner": "123456789012",
      "Endpoint": "my-email@example.com",
      "Protocol": "email",
      "TopicArn": "arn:aws:sns:us-west-2:123456789012:my-topic",
      "SubscriptionArn": "arn:aws:sns:us-west-2:123456789012:my-
topic:8a21d249-4329-4871-acc6-7be709c6ea7f"
    }
  ]
}
```

- Para obter detalhes da API, consulte [ListSubscriptions](#) em Referência de AWS CLI Comandos.

## Java

## SDK para Java 2.x

 Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.sns.SnsClient;
import software.amazon.awssdk.services.sns.model.ListSubscriptionsRequest;
import software.amazon.awssdk.services.sns.model.ListSubscriptionsResponse;
import software.amazon.awssdk.services.sns.model.SnsException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
 * started.html
 */
public class ListSubscriptions {
    public static void main(String[] args) {
        SnsClient snsClient = SnsClient.builder()
            .region(Region.US_EAST_1)
            .build();

        listSNSSubscriptions(snsClient);
        snsClient.close();
    }

    public static void listSNSSubscriptions(SnsClient snsClient) {
        try {
            ListSubscriptionsRequest request = ListSubscriptionsRequest.builder()
                .build();

            ListSubscriptionsResponse result =
snsClient.listSubscriptions(request);
```

```
        System.out.println(result.subscriptions());

    } catch (SnsException e) {

        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}
}
```

- Para obter detalhes da API, consulte [ListSubscriptions](#) na Referência AWS SDK for Java 2.x da API.

## JavaScript

### SDK para JavaScript (v3)

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

Crie o cliente em um módulo separado e exporte-o.

```
import { SNSClient } from "@aws-sdk/client-sns";

// The AWS Region can be provided here using the `region` property. If you leave
// it blank
// the SDK will default to the region set in your AWS config.
export const snsClient = new SNSClient({});
```

Importe o SDK e os módulos do cliente e chame a API.

```
import { ListSubscriptionsByTopicCommand } from "@aws-sdk/client-sns";
import { snsClient } from "../libs/snsClient.js";

/**
```

```
* @param {string} topicArn - The ARN of the topic for which you wish to list
subscriptions.
*/
export const listSubscriptionsByTopic = async (topicArn = "TOPIC_ARN") => {
  const response = await snsClient.send(
    new ListSubscriptionsByTopicCommand({ TopicArn: topicArn }),
  );

  console.log(response);
  // {
  //   '$metadata': {
  //     httpStatusCode: 200,
  //     requestId: '0934fedf-0c4b-572e-9ed2-a3e38fadb0c8',
  //     extendedRequestId: undefined,
  //     cfId: undefined,
  //     attempts: 1,
  //     totalRetryDelay: 0
  //   },
  //   Subscriptions: [
  //     {
  //       SubscriptionArn: 'PendingConfirmation',
  //       Owner: '901487484989',
  //       Protocol: 'email',
  //       Endpoint: 'corepyle@amazon.com',
  //       TopicArn: 'arn:aws:sns:us-east-1:901487484989:mytopic'
  //     }
  //   ]
  // }
  return response;
};
```

- Para obter mais informações, consulte o [Guia do desenvolvedor do AWS SDK for JavaScript](#).
- Para obter detalhes da API, consulte [ListSubscriptions](#) na Referência AWS SDK for JavaScript da API.

## Kotlin

### SDK para Kotlin

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
suspend fun listSNSSubscriptions() {  
  
    SnsClient { region = "us-east-1" }.use { snsClient ->  
        val response = snsClient.listSubscriptions(ListSubscriptionsRequest {})  
        response.subscriptions?.forEach { sub ->  
            println("Sub ARN is ${sub.subscriptionArn}")  
            println("Sub protocol is ${sub.protocol}")  
        }  
    }  
}
```

- Para obter detalhes da API, consulte a [ListSubscriptions](#) referência da API AWS SDK for Kotlin.

## PHP

### SDK para PHP

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
require 'vendor/autoload.php';  
  
use Aws\Exception\AwsException;  
use Aws\Sns\SnsClient;
```

```
/**
 * Returns a list of Amazon SNS subscriptions in the requested region.
 *
 * This code expects that you have AWS credentials set up per:
 * https://docs.aws.amazon.com/sdk-for-php/v3/developer-guide/
guide_credentials.html
 */

$SnSClient = new SnsClient([
    'profile' => 'default',
    'region' => 'us-east-1',
    'version' => '2010-03-31'
]);

try {
    $result = $SnSClient->listSubscriptions();
    var_dump($result);
} catch (AwsException $e) {
    // output error message if fails
    error_log($e->getMessage());
}
```

- Para obter detalhes da API, consulte [ListSubscriptions](#) na Referência AWS SDK for PHP da API.

## Python

### SDK para Python (Boto3)

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
class SnsWrapper:
    """Encapsulates Amazon SNS topic and subscription functions."""
```

```
def __init__(self, sns_resource):
    """
    :param sns_resource: A Boto3 Amazon SNS resource.
    """
    self.sns_resource = sns_resource

def list_subscriptions(self, topic=None):
    """
    Lists subscriptions for the current account, optionally limited to a
    specific topic.

    :param topic: When specified, only subscriptions to this topic are
    returned.
    :return: An iterator that yields the subscriptions.
    """
    try:
        if topic is None:
            subs_iter = self.sns_resource.subscriptions.all()
        else:
            subs_iter = topic.subscriptions.all()
            logger.info("Got subscriptions.")
    except ClientError:
        logger.exception("Couldn't get subscriptions.")
        raise
    else:
        return subs_iter
```

- Para obter detalhes da API, consulte a [ListSubscriptions](#) Referência da API AWS SDK for Python (Boto3).

## Ruby

### SDK para Ruby

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
# This class demonstrates how to list subscriptions to an Amazon Simple
Notification Service (SNS) topic
class SnsSubscriptionLister
  def initialize(sns_client)
    @sns_client = sns_client
    @logger = Logger.new($stdout)
  end

  # Lists subscriptions for a given SNS topic
  # @param topic_arn [String] The ARN of the SNS topic
  # @return [Types::ListSubscriptionsResponse] subscriptions: The response object
  def list_subscriptions(topic_arn)
    @logger.info("Listing subscriptions for topic: #{topic_arn}")
    subscriptions = @sns_client.list_subscriptions_by_topic(topic_arn: topic_arn)
    subscriptions.subscriptions.each do |subscription|
      @logger.info("Subscription endpoint: #{subscription.endpoint}")
    end
    subscriptions
  rescue Aws::SNS::Errors::ServiceError => e
    @logger.error("Error listing subscriptions: #{e.message}")
    raise
  end
end

# Example usage:
if $PROGRAM_NAME == __FILE__
  sns_client = Aws::SNS::Client.new
  topic_arn = "SNS_TOPIC_ARN" # Replace with your SNS topic ARN
  lister = SnsSubscriptionLister.new(sns_client)

  begin
    lister.list_subscriptions(topic_arn)
  rescue StandardError => e
    puts "Failed to list subscriptions: #{e.message}"
    exit 1
  end
end
```

- Para obter mais informações, consulte o [Guia do desenvolvedor do AWS SDK for Ruby](#).
- Para obter detalhes da API, consulte [ListSubscriptions](#) a Referência AWS SDK for Ruby da API.

## SAP ABAP

### SDK para SAP ABAP

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
TRY.  
    oo_result = lo_sns->listsubscriptions( ).           " oo_result is  
returned for testing purposes. "  
    DATA(lt_subscriptions) = oo_result->get_subscriptions( ).  
    MESSAGE 'Retrieved list of subscribers.' TYPE 'I'.  
    CATCH /aws1/cx_rt_generic.  
    MESSAGE 'Unable to list subscribers.' TYPE 'E'.  
ENDTRY.
```

- Para obter detalhes da API, consulte a [ListSubscriptions](#) referência da API AWS SDK for SAP ABAP.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulte [Usando o Amazon SNS com um SDK AWS](#). Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

## Use **ListTopics** com um AWS SDK ou CLI

Os exemplos de códigos a seguir mostram como usar `ListTopics`.

### .NET

#### AWS SDK for .NET

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
using System;
using System.Collections.Generic;
using System.Threading.Tasks;
using Amazon.SimpleNotificationService;
using Amazon.SimpleNotificationService.Model;

/// <summary>
/// Lists the Amazon Simple Notification Service (Amazon SNS)
/// topics for the current account.
/// </summary>
public class ListSNSTopics
{
    public static async Task Main()
    {
        IAmazonSimpleNotificationService client = new
AmazonSimpleNotificationServiceClient();

        await GetTopicListAsync(client);
    }

    /// <summary>
    /// Retrieves the list of Amazon SNS topics in groups of up to 100
    /// topics.
    /// </summary>
    /// <param name="client">The initialized Amazon SNS client object used
    /// to retrieve the list of topics.</param>
    public static async Task
GetTopicListAsync(IAmazonSimpleNotificationService client)
    {
        // If there are more than 100 Amazon SNS topics, the call to
        // ListTopicsAsync will return a value to pass to the
        // method to retrieve the next 100 (or less) topics.
        string nextToken = string.Empty;

        do
        {
            var response = await client.ListTopicsAsync(nextToken);
            DisplayTopicsList(response.Topics);
            nextToken = response.NextToken;
        }
        while (!string.IsNullOrEmpty(nextToken));
    }
}
```

```

    /// <summary>
    /// Displays the list of Amazon SNS Topic ARNs.
    /// </summary>
    /// <param name="topicList">The list of Topic ARNs.</param>
    public static void DisplayTopicsList(List<Topic> topicList)
    {
        foreach (var topic in topicList)
        {
            Console.WriteLine($"{topic.TopicArn}");
        }
    }
}

```

- Para obter detalhes da API, consulte [ListTopics](#) na Referência AWS SDK for .NET da API.

## C++

### SDK para C++

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```

/*! Retrieve a list of Amazon Simple Notification Service (Amazon SNS) topics.
 *!
 *! \param clientConfiguration: AWS client configuration.
 *! \return bool: Function succeeded.
 *! */
bool
AwsDoc::SNS::listTopics(const Aws::Client::ClientConfiguration
&clientConfiguration) {
    Aws::SNS::SNSClient snsClient(clientConfiguration);

    Aws::String nextToken; // Next token is used to handle a paginated response.
    bool result = true;
    do {
        Aws::SNS::Model::ListTopicsRequest request;

```

```
    if (!nextToken.empty()) {
        request.SetNextToken(nextToken);
    }

    const Aws::SNS::Model::ListTopicsOutcome outcome = snsClient.ListTopics(
        request);

    if (outcome.IsSuccess()) {
        std::cout << "Topics list:" << std::endl;
        for (auto const &topic: outcome.GetResult().GetTopics()) {
            std::cout << " * " << topic.GetTopicArn() << std::endl;
        }
    }
    else {
        std::cerr << "Error listing topics " <<
outcome.GetError().GetMessage() <<
            std::endl;
        result = false;
        break;
    }

    nextToken = outcome.GetResult().GetNextToken();
} while (!nextToken.empty());

return result;
}
```

- Para obter detalhes da API, consulte [ListTopics](#) na Referência AWS SDK for C++ da API.

## CLI

### AWS CLI

Listar os tópicos do SNS

O `list-topics` exemplo a seguir lista todos os tópicos do SNS em sua AWS conta.

```
aws sns list-topics
```

Saída:

```
{
```

```
"Topics": [  
  {  
    "TopicArn": "arn:aws:sns:us-west-2:123456789012:my-topic"  
  }  
]  
}
```

- Para obter detalhes da API, consulte [ListTopics](#) em Referência de AWS CLI Comandos.

Go

## SDK para Go V2

### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
package main  
  
import (  
  "context"  
  "fmt"  
  "log"  
  
  "github.com/aws/aws-sdk-go-v2/config"  
  "github.com/aws/aws-sdk-go-v2/service/sns"  
  "github.com/aws/aws-sdk-go-v2/service/sns/types"  
)  
  
// main uses the AWS SDK for Go V2 to create an Amazon Simple Notification  
// Service  
// (Amazon SNS) client and list the topics in your account.  
// This example uses the default settings specified in your shared credentials  
// and config files.  
func main() {  
  sdkConfig, err := config.LoadDefaultConfig(context.TODO())  
  if err != nil {  
    fmt.Println("Couldn't load default configuration. Have you set up your AWS  
account?")  
  }  
}
```



```
import software.amazon.awssdk.services.sns.model.ListTopicsResponse;
import software.amazon.awssdk.services.sns.model.SnsException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
 * started.html
 */
public class ListTopics {
    public static void main(String[] args) {
        SnsClient snsClient = SnsClient.builder()
            .region(Region.US_EAST_1)
            .build();

        listSNSTopics(snsClient);
        snsClient.close();
    }

    public static void listSNSTopics(SnsClient snsClient) {
        try {
            ListTopicsRequest request = ListTopicsRequest.builder()
                .build();

            ListTopicsResponse result = snsClient.listTopics(request);
            System.out.println(
                "Status was " + result.sdkHttpResponse().statusCode() + "\n
\nTopics\n\n" + result.topics());

        } catch (SnsException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Para obter detalhes da API, consulte [ListTopics](#) Referência AWS SDK for Java 2.x da API.

## JavaScript

### SDK para JavaScript (v3)

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

Crie o cliente em um módulo separado e exporte-o.

```
import { SNSClient } from "@aws-sdk/client-sns";

// The AWS Region can be provided here using the `region` property. If you leave
// it blank
// the SDK will default to the region set in your AWS config.
export const snsClient = new SNSClient({});
```

Importe o SDK e os módulos do cliente e chame a API.

```
import { ListTopicsCommand } from "@aws-sdk/client-sns";
import { snsClient } from "../libs/snsClient.js";

export const listTopics = async () => {
  const response = await snsClient.send(new ListTopicsCommand({}));
  console.log(response);
  // {
  //   '$metadata': {
  //     httpStatusCode: 200,
  //     requestId: '936bc5ad-83ca-53c2-b0b7-9891167b909e',
  //     extendedRequestId: undefined,
  //     cfId: undefined,
  //     attempts: 1,
  //     totalRetryDelay: 0
  //   },
  //   Topics: [ { TopicArn: 'arn:aws:sns:us-east-1:xxxxxxxxxxxx:mytopic' } ]
  // }
  return response;
};
```

- Para obter mais informações, consulte o [Guia do desenvolvedor do AWS SDK for JavaScript](#).
- Para obter detalhes da API, consulte [ListTopics](#) na Referência AWS SDK for JavaScript da API.

## Kotlin

### SDK para Kotlin

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
suspend fun listSNSTopics() {  
  
    SnsClient { region = "us-east-1" }.use { snsClient ->  
        val response = snsClient.listTopics(ListTopicsRequest { })  
        response.topics?.forEach { topic ->  
            println("The topic ARN is ${topic.topicArn}")  
        }  
    }  
}
```

- Para obter detalhes da API, consulte a [ListTopics](#) referência da API AWS SDK for Kotlin.

## PHP

### SDK para PHP

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
require 'vendor/autoload.php';

use Aws\Exception\AwsException;
use Aws\Sns\SnsClient;

/**
 * Returns a list of the requester's topics from your AWS SNS account in the
 * region specified.
 *
 * This code expects that you have AWS credentials set up per:
 * https://docs.aws.amazon.com/sdk-for-php/v3/developer-guide/
 * guide_credentials.html
 */

$SnSClient = new SnsClient([
    'profile' => 'default',
    'region' => 'us-east-1',
    'version' => '2010-03-31'
]);

try {
    $result = $SnSClient->listTopics();
    var_dump($result);
} catch (AwsException $e) {
    // output error message if fails
    error_log($e->getMessage());
}
```

- Para obter detalhes da API, consulte [ListTopics](#) na Referência AWS SDK for PHP da API.

## Python

### SDK para Python (Boto3)

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
class SnsWrapper:
    """Encapsulates Amazon SNS topic and subscription functions."""

    def __init__(self, sns_resource):
        """
        :param sns_resource: A Boto3 Amazon SNS resource.
        """
        self.sns_resource = sns_resource

    def list_topics(self):
        """
        Lists topics for the current account.

        :return: An iterator that yields the topics.
        """
        try:
            topics_iter = self.sns_resource.topics.all()
            logger.info("Got topics.")
        except ClientError:
            logger.exception("Couldn't get topics.")
            raise
        else:
            return topics_iter
```

- Para obter detalhes da API, consulte a [ListTopics](#) Referência da API AWS SDK for Python (Boto3).

## Ruby

### SDK para Ruby

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
require "aws-sdk-sns" # v2: require 'aws-sdk'

def list_topics?(sns_client)
  sns_client.topics.each do |topic|
    puts topic.arn
  rescue StandardError => e
    puts "Error while listing the topics: #{e.message}"
  end
end

def run_me

  region = "REGION"
  sns_client = Aws::SNS::Resource.new(region: region)

  puts "Listing the topics."

  if list_topics?(sns_client)
  else
    puts "The bucket was not created. Stopping program."
    exit 1
  end
end

# Example usage:
run_me if $PROGRAM_NAME == __FILE__
```

- Para obter mais informações, consulte o [Guia do desenvolvedor do AWS SDK for Ruby](#).
- Para obter detalhes da API, consulte [ListTopics](#) a Referência AWS SDK for Ruby da API.

## Rust

### SDK para Rust

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```

async fn show_topics(client: &Client) -> Result<(), Error> {
    let resp = client.list_topics().send().await?;

    println!("Topic ARNs:");

    for topic in resp.topics() {
        println!("{}", topic.topic_arn().unwrap_or_default());
    }

    Ok(())
}

```

- Para obter detalhes da API, consulte a [ListTopics](#) referência da API AWS SDK for Rust.

## SAP ABAP

### SDK para SAP ABAP

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```

TRY.
    oo_result = lo_sns->listtopics( ).           " oo_result is returned for
testing purposes. "
    DATA(lt_topics) = oo_result->get_topics( ).
    MESSAGE 'Retrieved list of topics.' TYPE 'I'.
CATCH /aws1/cx_rt_generic.
    MESSAGE 'Unable to list topics.' TYPE 'E'.
ENDTRY.

```

- Para obter detalhes da API, consulte a [ListTopics](#) referência da API AWS SDK for SAP ABAP.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulte [Usando o Amazon SNS com um SDK AWS](#). Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

## Use **Publish** com um AWS SDK ou CLI

Os exemplos de códigos a seguir mostram como usar Publish.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto nos seguintes exemplos de código:

- [Criar e publicar em um tópico FIFO](#)
- [Publicar uma mensagem de texto SMS](#)
- [Publicar mensagens em filas](#)

.NET

AWS SDK for .NET

### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

Publique uma mensagem em um tópico.

```
using System;
using System.Threading.Tasks;
using Amazon.SimpleNotificationService;
using Amazon.SimpleNotificationService.Model;

/// <summary>
/// This example publishes a message to an Amazon Simple Notification
/// Service (Amazon SNS) topic.
/// </summary>
public class PublishToSNSTopic
{
    public static async Task Main()
    {
```

```
        string topicArn = "arn:aws:sns:us-
east-2:000000000000:ExampleSNSTopic";
        string messageText = "This is an example message to publish to the
ExampleSNSTopic.";

        IAmazonSimpleNotificationService client = new
AmazonSimpleNotificationServiceClient();

        await PublishToTopicAsync(client, topicArn, messageText);
    }

    /// <summary>
    /// Publishes a message to an Amazon SNS topic.
    /// </summary>
    /// <param name="client">The initialized client object used to publish
    /// to the Amazon SNS topic.</param>
    /// <param name="topicArn">The ARN of the topic.</param>
    /// <param name="messageText">The text of the message.</param>
    public static async Task PublishToTopicAsync(
        IAmazonSimpleNotificationService client,
        string topicArn,
        string messageText)
    {
        var request = new PublishRequest
        {
            TopicArn = topicArn,
            Message = messageText,
        };

        var response = await client.PublishAsync(request);

        Console.WriteLine($"Successfully published message ID:
{response.MessageId}");
    }
}
```

Publique uma mensagem em um tópico com opções de grupo, duplicação e atributo.

```
/// <summary>
/// Publish messages using user settings.
/// </summary>
```

```
/// <returns>Async task.</returns>
public static async Task PublishMessages()
{
    Console.WriteLine("Now we can publish messages.");

    var keepSendingMessages = true;
    string? deduplicationId = null;
    string? toneAttribute = null;
    while (keepSendingMessages)
    {
        Console.WriteLine();
        var message = GetUserResponse("Enter a message to publish.", "This is
a sample message");

        if (_useFifoTopic)
        {
            Console.WriteLine("Because you are using a FIFO topic, you must
set a message group ID." +
                "\r\nAll messages within the same group will be
received in the order " +
                "they were published.");

            Console.WriteLine();
            var messageId = GetUserResponse("Enter a message group ID
for this message:", "1");

            if (!_useContentBasedDeduplication)
            {
                Console.WriteLine("Because you are not using content-based
deduplication, " +
                    "you must enter a deduplication ID.");

                Console.WriteLine("Enter a deduplication ID for this
message.");
                deduplicationId = GetUserResponse("Enter a deduplication ID
for this message.", "1");
            }

            if (GetYesNoResponse("Add an attribute to this message?"))
            {
                Console.WriteLine("Enter a number for an attribute.");
                for (int i = 0; i < _tones.Length; i++)
                {
                    Console.WriteLine($"{i + 1}. {_tones[i]}");
                }
            }
        }
    }
}
```

```

        }

        var selection = GetUserResponse("", "1");
        int.TryParse(selection, out var selectionNumber);

        if (selectionNumber > 0 && selectionNumber < _tones.Length)
        {
            toneAttribute = _tones[selectionNumber - 1];
        }
    }

    var messageID = await SnsWrapper.PublishToTopicWithAttribute(
        _topicArn, message, "tone", toneAttribute, deduplicationId,
messageGroupId);

    Console.WriteLine($"Message published with id {messageID}.");
}

keepSendingMessages = GetYesNoResponse("Send another message?",
false);
}
}

```

Aplice as seleções do usuário à ação de publicação.

```

    /// <summary>
    /// Publish a message to a topic with an attribute and optional deduplication
and group IDs.
    /// </summary>
    /// <param name="topicArn">The ARN of the topic.</param>
    /// <param name="message">The message to publish.</param>
    /// <param name="attributeName">The optional attribute for the message.</
param>
    /// <param name="attributeValue">The optional attribute value for the
message.</param>
    /// <param name="deduplicationId">The optional deduplication ID for the
message.</param>
    /// <param name="groupId">The optional group ID for the message.</param>
    /// <returns>The ID of the message published.</returns>
    public async Task<string> PublishToTopicWithAttribute(
        string topicArn,
        string message,

```

```
string? attributeName = null,
string? attributeValue = null,
string? deduplicationId = null,
string? groupId = null)
{
    var publishRequest = new PublishRequest()
    {
        TopicArn = topicArn,
        Message = message,
        MessageDeduplicationId = deduplicationId,
        MessageGroupId = groupId
    };

    if (attributeValue != null)
    {
        // Add the string attribute if it exists.
        publishRequest.MessageAttributes =
            new Dictionary<string, MessageAttributeValue>
            {
                { attributeName!, new MessageAttributeValue() { StringValue =
attributeValue, DataType = "String" } }
            };
    }

    var publishResponse = await
_amazonSNSClient.PublishAsync(publishRequest);
    return publishResponse.MessageId;
}
```

- Para obter detalhes da API, consulte [Publish](#) na Referência da API AWS SDK for .NET .

## C++

### SDK para C++

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```

//! Send a message to an Amazon Simple Notification Service (Amazon SNS) topic.
/*!
  \param message: The message to publish.
  \param topicARN: The Amazon Resource Name (ARN) for an Amazon SNS topic.
  \param clientConfiguration: AWS client configuration.
  \return bool: Function succeeded.
*/
bool AwsDoc::SNS::publishToTopic(const Aws::String &message,
                                const Aws::String &topicARN,
                                const Aws::Client::ClientConfiguration
&clientConfiguration) {
    Aws::SNS::SNSClient snsClient(clientConfiguration);

    Aws::SNS::Model::PublishRequest request;
    request.SetMessage(message);
    request.SetTopicArn(topicARN);

    const Aws::SNS::Model::PublishOutcome outcome = snsClient.Publish(request);

    if (outcome.IsSuccess()) {
        std::cout << "Message published successfully with id '"
                  << outcome.GetResult().GetMessageId() << "'." << std::endl;
    }
    else {
        std::cerr << "Error while publishing message "
                  << outcome.GetError().GetMessage()
                  << std::endl;
    }

    return outcome.IsSuccess();
}

```

Publique uma mensagem com um atributo.

```

static const Aws::String TONE_ATTRIBUTE("tone");
static const Aws::Vector<Aws::String> TONES = {"cheerful", "funny",
"serious",
                                                "sincere"};

Aws::Client::ClientConfiguration clientConfig;
// Optional: Set to the AWS Region (overrides config file).
// clientConfig.region = "us-east-1";

```

```
Aws::SNS::SNSClient snsClient(clientConfiguration);

Aws::SNS::Model::PublishRequest request;
request.SetTopicArn(topicARN);
Aws::String message = askQuestion("Enter a message text to publish. ");
request.SetMessage(message);

if (filteringMessages && askYesNoQuestion(
    "Add an attribute to this message? (y/n) ")) {
    for (size_t i = 0; i < TONES.size(); ++i) {
        std::cout << " " << (i + 1) << ". " << TONES[i] << std::endl;
    }
    int selection = askQuestionForIntRange(
        "Enter a number for an attribute. ",
        1, static_cast<int>(TONES.size()));
    Aws::SNS::Model::MessageAttributeValue messageAttributeValue;
    messageAttributeValue.SetDataType("String");
    messageAttributeValue.SetStringValue(TONES[selection - 1]);
    request.AddMessageAttributes(TONE_ATTRIBUTE, messageAttributeValue);
}

Aws::SNS::Model::PublishOutcome outcome = snsClient.Publish(request);

if (outcome.IsSuccess()) {
    std::cout << "Your message was successfully published." << std::endl;
}
else {
    std::cerr << "Error with TopicsAndQueues::Publish. "
        << outcome.GetError().GetMessage()
        << std::endl;

    cleanUp(topicARN,
        queueURLS,
        subscriptionARNS,
        snsClient,
        sqsClient);

    return false;
}
```

- Para obter detalhes da API, consulte [Publish](#) na Referência da API AWS SDK for C++ .

## CLI

## AWS CLI

Exemplo 1: Para publicar uma mensagem em um tópico:

O exemplo `publish` a seguir publica a mensagem específica no tópico do SNS especificado. A mensagem é proveniente de um arquivo de texto, o que permite incluir quebras de linha.

```
aws sns publish \  
  --topic-arn "arn:aws:sns:us-west-2:123456789012:my-topic" \  
  --message file://message.txt
```

Conteúdo de `message.txt`:

```
Hello World  
Second Line
```

Saída:

```
{  
  "MessageId": "123a45b6-7890-12c3-45d6-111122223333"  
}
```

Exemplo 2: Para publicar uma mensagem SMS em um número de telefone

O exemplo `publish` a seguir publica a mensagem `Hello world!` no número de telefone `+1-555-555-0100`.

```
aws sns publish \  
  --message "Hello world!" \  
  --phone-number +1-555-555-0100
```

Saída:

```
{  
  "MessageId": "123a45b6-7890-12c3-45d6-333322221111"  
}
```

- Para obter detalhes da API, consulte [Publish](#) na Referência de comandos da AWS CLI .

## Go

## SDK para Go V2

 Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
// SnsActions encapsulates the Amazon Simple Notification Service (Amazon SNS)
// actions
// used in the examples.
type SnsActions struct {
    SnsClient *sns.Client
}

// Publish publishes a message to an Amazon SNS topic. The message is then sent
// to all
// subscribers. When the topic is a FIFO topic, the message must also contain a
// group ID
// and, when ID-based deduplication is used, a deduplication ID. An optional key-
// value
// filter attribute can be specified so that the message can be filtered
// according to
// a filter policy.
func (actor SnsActions) Publish(topicArn string, message string, groupId string,
    dedupId string, filterKey string, filterValue string) error {
    publishInput := sns.PublishInput{TopicArn: aws.String(topicArn), Message:
    aws.String(message)}
    if groupId != "" {
        publishInput.MessageGroupId = aws.String(groupId)
    }
    if dedupId != "" {
        publishInput.MessageDeduplicationId = aws.String(dedupId)
    }
    if filterKey != "" && filterValue != "" {
        publishInput.MessageAttributes = map[string]types.MessageAttributeValue{
```

```
    filterKey: {DataType: aws.String("String"), StringValue:
aws.String(filterValue)},
    }
}
_, err := actor.SnsClient.Publish(context.TODO(), &publishInput)
if err != nil {
    log.Printf("Couldn't publish message to topic %v. Here's why: %v", topicArn,
err)
}
return err
}
```

- Para obter detalhes da API, consulte [Publish](#) na Referência da API AWS SDK for Go .

## Java

### SDK para Java 2.x

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.sns.SnsClient;
import software.amazon.awssdk.services.sns.model.PublishRequest;
import software.amazon.awssdk.services.sns.model.PublishResponse;
import software.amazon.awssdk.services.sns.model.SnsException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
 * started.html
 */
public class PublishTopic {
```

```
public static void main(String[] args) {
    final String usage = ""

        Usage:    <message> <topicArn>

        Where:
            message - The message text to send.
            topicArn - The ARN of the topic to publish.
        """;

    if (args.length != 2) {
        System.out.println(usage);
        System.exit(1);
    }

    String message = args[0];
    String topicArn = args[1];
    SnsClient snsClient = SnsClient.builder()
        .region(Region.US_EAST_1)
        .build();
    pubTopic(snsClient, message, topicArn);
    snsClient.close();
}

public static void pubTopic(SnsClient snsClient, String message, String
topicArn) {
    try {
        PublishRequest request = PublishRequest.builder()
            .message(message)
            .topicArn(topicArn)
            .build();

        PublishResponse result = snsClient.publish(request);
        System.out
            .println(result.messageId() + " Message sent. Status is " +
result.sdkHttpResponse().statusCode());

    } catch (SnsException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}
}
```

- Para obter detalhes da API, consulte [Publish](#) na Referência da API AWS SDK for Java 2.x .

## JavaScript

### SDK para JavaScript (v3)

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

Crie o cliente em um módulo separado e exporte-o.

```
import { SNSClient } from "@aws-sdk/client-sns";

// The AWS Region can be provided here using the `region` property. If you leave
// it blank
// the SDK will default to the region set in your AWS config.
export const snsClient = new SNSClient({});
```

Importe o SDK e os módulos do cliente e chame a API.

```
import { PublishCommand } from "@aws-sdk/client-sns";
import { snsClient } from "../libs/snsClient.js";

/**
 * @param {string | Record<string, any>} message - The message to send. Can be a
 * plain string or an object
 *
 * if you are using the `json`
 * `MessageStructure`.
 * @param {string} topicArn - The ARN of the topic to which you would like to
 * publish.
 */
export const publish = async (
  message = "Hello from SNS!",
  topicArn = "TOPIC_ARN",
) => {
```

```
const response = await snsClient.send(
  new PublishCommand({
    Message: message,
    TopicArn: topicArn,
  }),
);
console.log(response);
// {
//   '$metadata': {
//     httpStatusCode: 200,
//     requestId: 'e7f77526-e295-5325-9ee4-281a43ad1f05',
//     extendedRequestId: undefined,
//     cfId: undefined,
//     attempts: 1,
//     totalRetryDelay: 0
//   },
//   MessageId: 'xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxxxxx'
// }
return response;
};
```

Publique uma mensagem em um tópico com opções de grupo, duplicação e atributo.

```
async publishMessages() {
  const message = await this.prompter.input({
    message: MESSAGES.publishMessagePrompt,
  });

  let groupId, deduplicationId, choices;

  if (this.isFifo) {
    await this.logger.log(MESSAGES.groupIdNotice);
    groupId = await this.prompter.input({
      message: MESSAGES.groupIdPrompt,
    });
  }

  if (this.autoDedup === false) {
    await this.logger.log(MESSAGES.deduplicationIdNotice);
    deduplicationId = await this.prompter.input({
      message: MESSAGES.deduplicationIdPrompt,
    });
  }
}
```

```
    choices = await this.prompter.checkbox({
      message: MESSAGES.messageAttributesPrompt,
      choices: toneChoices,
    });
  }

  await this.snsClient.send(
    new PublishCommand({
      TopicArn: this.topicArn,
      Message: message,
      ...(groupId
        ? {
            MessageGroupId: groupId,
          }
        : {}),
      ...(deduplicationId
        ? {
            MessageDeduplicationId: deduplicationId,
          }
        : {}),
      ...(choices
        ? {
            MessageAttributes: {
              tone: {
                DataType: "String.Array",
                StringValue: JSON.stringify(choices),
              },
            },
          }
        : {}),
    })),
  );

  const publishAnother = await this.prompter.confirm({
    message: MESSAGES.publishAnother,
  });

  if (publishAnother) {
    await this.publishMessages();
  }
}
```

- Para obter mais informações, consulte o [Guia do desenvolvedor do AWS SDK for JavaScript](#).
- Para obter detalhes da API, consulte [Publish](#) na Referência da API AWS SDK for JavaScript

## Kotlin

### SDK para Kotlin

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
suspend fun pubTopic(topicArnVal: String, messageVal: String) {  
  
    val request = PublishRequest {  
        message = messageVal  
        topicArn = topicArnVal  
    }  
  
    SnsClient { region = "us-east-1" }.use { snsClient ->  
        val result = snsClient.publish(request)  
        println("${result.messageId} message sent.")  
    }  
}
```

- Para obter detalhes da API, consulte [Publish](#) na Referência da API AWS SDK para Kotlin.

## PHP

## SDK para PHP

 Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
require 'vendor/autoload.php';

use Aws\Exception\AwsException;
use Aws\Sns\SnsClient;

/**
 * Sends a message to an Amazon SNS topic.
 *
 * This code expects that you have AWS credentials set up per:
 * https://docs.aws.amazon.com/sdk-for-php/v3/developer-guide/
 * guide_credentials.html
 */

$SnSClient = new SnsClient([
    'profile' => 'default',
    'region' => 'us-east-1',
    'version' => '2010-03-31'
]);

$message = 'This message is sent from a Amazon SNS code sample.';
$topic = 'arn:aws:sns:us-east-1:111122223333:MyTopic';

try {
    $result = $SnSClient->publish([
        'Message' => $message,
        'TopicArn' => $topic,
    ]);
    var_dump($result);
} catch (AwsException $e) {
    // output error message if fails
    error_log($e->getMessage());
}
```

```
}
```

- Para obter mais informações, consulte o [Guia do desenvolvedor do AWS SDK for PHP](#).
- Para obter detalhes da API, consulte [Publish](#) na Referência da API AWS SDK for PHP .

## PowerShell

### Ferramentas para PowerShell

Exemplo 1: Este exemplo mostra a publicação de uma mensagem com uma única `MessageAttribute` declaração em linha.

```
Publish-SNSMessage -TopicArn "arn:aws:sns:us-west-2:123456789012:my-topic" -  
Message "Hello" -MessageAttribute  
@{'City'=[Amazon.SimpleNotificationService.Model.MessageAttributeValue]@{DataType='String'  
StringValue = 'AnyCity'}}
```

Exemplo 2: Este exemplo mostra a publicação de uma mensagem com várias `MessageAttributes` declaradas com antecedência.

```
$cityAttributeValue = New-Object  
Amazon.SimpleNotificationService.Model.MessageAttributeValue  
$cityAttributeValue.DataType = "String"  
$cityAttributeValue.StringValue = "AnyCity"  
  
$populationAttributeValue = New-Object  
Amazon.SimpleNotificationService.Model.MessageAttributeValue  
$populationAttributeValue.DataType = "Number"  
$populationAttributeValue.StringValue = "1250800"  
  
$messageAttributes = New-Object System.Collections.Hashtable  
$messageAttributes.Add("City", $cityAttributeValue)  
$messageAttributes.Add("Population", $populationAttributeValue)  
  
Publish-SNSMessage -TopicArn "arn:aws:sns:us-west-2:123456789012:my-topic" -  
Message "Hello" -MessageAttribute $messageAttributes
```

- Para obter detalhes da API, consulte [Publicar](#) no AWS Tools for PowerShell Cmdlet Reference.

## Python

### SDK para Python (Boto3)

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

Publique uma mensagem com atributos para que uma assinatura possa filtrar com base em atributos.

```
class SnsWrapper:
    """Encapsulates Amazon SNS topic and subscription functions."""

    def __init__(self, sns_resource):
        """
        :param sns_resource: A Boto3 Amazon SNS resource.
        """
        self.sns_resource = sns_resource

    @staticmethod
    def publish_message(topic, message, attributes):
        """
        Publishes a message, with attributes, to a topic. Subscriptions can be
        filtered
        based on message attributes so that a subscription receives messages only
        when specified attributes are present.

        :param topic: The topic to publish to.
        :param message: The message to publish.
        :param attributes: The key-value attributes to attach to the message.
        Values
            must be either `str` or `bytes`.
        :return: The ID of the message.
        """
        try:
            att_dict = {}
            for key, value in attributes.items():
                if isinstance(value, str):
```

```

        att_dict[key] = {"DataType": "String", "StringValue": value}
    elif isinstance(value, bytes):
        att_dict[key] = {"DataType": "Binary", "BinaryValue": value}
    response = topic.publish(Message=message, MessageAttributes=att_dict)
    message_id = response["MessageId"]
    logger.info(
        "Published message with attributes %s to topic %s.",
        attributes,
        topic.arn,
    )
except ClientError:
    logger.exception("Couldn't publish message to topic %s.", topic.arn)
    raise
else:
    return message_id

```

Publique uma mensagem que assume diferentes formas com base no protocolo do assinante.

```

class SnsWrapper:
    """Encapsulates Amazon SNS topic and subscription functions."""

    def __init__(self, sns_resource):
        """
        :param sns_resource: A Boto3 Amazon SNS resource.
        """
        self.sns_resource = sns_resource

    @staticmethod
    def publish_multi_message(
        topic, subject, default_message, sms_message, email_message
    ):
        """
        Publishes a multi-format message to a topic. A multi-format message takes
        different forms based on the protocol of the subscriber. For example,
        an SMS subscriber might receive a short version of the message
        while an email subscriber could receive a longer version.

        :param topic: The topic to publish to.
        :param subject: The subject of the message.

```

```
is
    :param default_message: The default version of the message. This version
    sent to subscribers that have protocols that are
not
    otherwise specified in the structured message.
    :param sms_message: The version of the message sent to SMS subscribers.
    :param email_message: The version of the message sent to email
subscribers.
    :return: The ID of the message.
    """
    try:
        message = {
            "default": default_message,
            "sms": sms_message,
            "email": email_message,
        }
        response = topic.publish(
            Message=json.dumps(message), Subject=subject,
MessageStructure="json"
        )
        message_id = response["MessageId"]
        logger.info("Published multi-format message to topic %s.", topic.arn)
    except ClientError:
        logger.exception("Couldn't publish message to topic %s.", topic.arn)
        raise
    else:
        return message_id
```

- Para obter detalhes da API, consulte [Publish](#) na Referência da API AWS SDK for Python (Boto3).

## Ruby

### SDK para Ruby

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
# Service class for sending messages using Amazon Simple Notification Service
(SNS)
class SnsMessageSender
  # Initializes the SnsMessageSender with an SNS client
  #
  # @param sns_client [Aws::SNS::Client] The SNS client
  def initialize(sns_client)
    @sns_client = sns_client
    @logger = Logger.new($stdout)
  end

  # Sends a message to a specified SNS topic
  #
  # @param topic_arn [String] The ARN of the SNS topic
  # @param message [String] The message to send
  # @return [Boolean] true if message was successfully sent, false otherwise
  def send_message(topic_arn, message)
    @sns_client.publish(topic_arn: topic_arn, message: message)
    @logger.info("Message sent successfully to #{topic_arn}.")
    true
  rescue Aws::SNS::Errors::ServiceError => e
    @logger.error("Error while sending the message: #{e.message}")
    false
  end
end

# Example usage:
if $PROGRAM_NAME == __FILE__
  topic_arn = "SNS_TOPIC_ARN" # Should be replaced with a real topic ARN
  message = "MESSAGE"        # Should be replaced with the actual message
  content
```

```
sns_client = Aws::SNS::Client.new
message_sender = SnsMessageSender.new(sns_client)

@logger.info("Sending message.")
unless message_sender.send_message(topic_arn, message)
  @logger.error("Message sending failed. Stopping program.")
  exit 1
end
end
```

- Para obter mais informações, consulte o [Guia do desenvolvedor do AWS SDK for Ruby](#).
- Para obter detalhes da API, consulte [Publish](#) na Referência da API AWS SDK for Ruby .

## Rust

### SDK for Rust

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
async fn subscribe_and_publish(
  client: &Client,
  topic_arn: &str,
  email_address: &str,
) -> Result<(), Error> {
  println!("Receiving on topic with ARN: `{}`", topic_arn);

  let rsp = client
    .subscribe()
    .topic_arn(topic_arn)
    .protocol("email")
    .endpoint(email_address)
    .send()
    .await?;

  println!("Added a subscription: {:?}", rsp);
}
```

```
let rsp = client
    .publish()
    .topic_arn(topic_arn)
    .message("hello sns!")
    .send()
    .await?;

println!("Published message: {:?}", rsp);

Ok(())
}
```

- Para obter detalhes da API, consulte [Publish](#) na Referência da API AWS SDK for Rust.

## SAP ABAP

### SDK para SAP ABAP

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
TRY.
    oo_result = lo_sns->publish(
testing purposes. "                                " oo_result is returned for
        iv_topicarn = iv_topic_arn
        iv_message = iv_message
    ).
    MESSAGE 'Message published to SNS topic.' TYPE 'I'.
CATCH /aws1/cx_snsnotfoundexception.
    MESSAGE 'Topic does not exist.' TYPE 'E'.
ENDTRY.
```

- Para obter os detalhes da API, consulte [Publicar](#) na Referência da API AWS SDK para SAP ABAP.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulte [Usando o Amazon SNS com um SDK AWS](#). Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

## Use **SetSMSAttributes** com um AWS SDK ou CLI

Os exemplos de códigos a seguir mostram como usar `SetSMSAttributes`.

C++

SDK para C++

### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

Como usar o Amazon SNS para definir o atributo `DefaultSMSType`.

```
#!/ Set the default settings for sending SMS messages.
/*!
 \param smsType: The type of SMS message that you will send by default.
 \param clientConfiguration: AWS client configuration.
 \return bool: Function succeeded.
 */
bool AwsDoc::SNS::setSMSType(const Aws::String &smsType,
                             const Aws::Client::ClientConfiguration
&clientConfiguration) {
    Aws::SNS::SNSClient snsClient(clientConfiguration);

    Aws::SNS::Model::SetSMSAttributesRequest request;
    request.AddAttributes("DefaultSMSType", smsType);

    const Aws::SNS::Model::SetSMSAttributesOutcome outcome =
snsClient.SetSMSAttributes(
    request);

    if (outcome.IsSuccess()) {
        std::cout << "SMS Type set successfully " << std::endl;
    }
    else {
```

```
        std::cerr << "Error while setting SMS Type: '"  
                << outcome.GetError().GetMessage()  
                << "'" << std::endl;  
    }  
  
    return outcome.IsSuccess();  
}
```

- Para receber detalhes da API, consulte [SetSMSAttributes](#) na Referência da API AWS SDK for C++ .

## CLI

### AWS CLI

Para definir atributos de mensagens SMS

O exemplo `set-sms-attributes` a seguir define o ID do remetente padrão para mensagens SMS como `MyName`.

```
aws sns set-sms-attributes \  
    --attributes DefaultSenderId=MyName
```

Este comando não produz saída.

- Para encontrar detalhes da API, consulte [SetSMSAttributes](#) na Referência de comandos da AWS CLI .

## Java

### SDK para Java 2.x

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
import software.amazon.awssdk.regions.Region;  
import software.amazon.awssdk.services.sns.SnsClient;
```

```
import software.amazon.awssdk.services.sns.model.SetSmsAttributesRequest;
import software.amazon.awssdk.services.sns.model.SetSmsAttributesResponse;
import software.amazon.awssdk.services.sns.model.SnsException;
import java.util.HashMap;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */
public class SetSMSAttributes {
    public static void main(String[] args) {
        HashMap<String, String> attributes = new HashMap<>(1);
        attributes.put("DefaultSMSType", "Transactional");
        attributes.put("UsageReportS3Bucket", "janbucket");

        SnsClient snsClient = SnsClient.builder()
            .region(Region.US_EAST_1)
            .build();
        setSMSAttributes(snsClient, attributes);
        snsClient.close();
    }

    public static void setSMSAttributes(SnsClient snsClient, HashMap<String,
String> attributes) {
        try {
            SetSmsAttributesRequest request = SetSmsAttributesRequest.builder()
                .attributes(attributes)
                .build();

            SetSmsAttributesResponse result =
snsClient.setSMSAttributes(request);
            System.out.println("Set default Attributes to " + attributes + ".
Status was "
                + result.sdkHttpResponse().statusCode());
        } catch (SnsException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

```
    }  
  }  
}
```

- Para receber detalhes da API, consulte [SetSMSAttributes](#) na Referência da API AWS SDK for Java 2.x .

## JavaScript

### SDK para JavaScript (v3)

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

Crie o cliente em um módulo separado e exporte-o.

```
import { SNSClient } from "@aws-sdk/client-sns";  
  
// The AWS Region can be provided here using the `region` property. If you leave  
// it blank  
// the SDK will default to the region set in your AWS config.  
export const snsClient = new SNSClient({});
```

Importe o SDK e os módulos do cliente e chame a API.

```
import { SetSMSAttributesCommand } from "@aws-sdk/client-sns";  
import { snsClient } from "../libs/snsClient.js";  
  
/**  
 * @param {"Transactional" | "Promotional"} defaultSmsType  
 */  
export const setSmsType = async (defaultSmsType = "Transactional") => {  
  const response = await snsClient.send(  
    new SetSMSAttributesCommand({  
      attributes: {  
        // Promotional - (Default) Noncritical messages, such as marketing  
        // messages.  

```

```
        // Transactional - Critical messages that support customer transactions,
        // such as one-time passcodes for multi-factor authentication.
        DefaultSMSType: defaultSmsType,
    },
 )),
);
console.log(response);
// {
//   '$metadata': {
//     httpStatusCode: 200,
//     requestId: '1885b977-2d7e-535e-8214-e44be727e265',
//     extendedRequestId: undefined,
//     cfId: undefined,
//     attempts: 1,
//     totalRetryDelay: 0
//   }
// }
return response;
};
```

- Para obter mais informações, consulte o [Guia do desenvolvedor do AWS SDK for JavaScript](#).
- Para receber detalhes da API, consulte [SetSMSAttributes](#) na Referência da API AWS SDK for JavaScript .

## PHP

### SDK para PHP

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
$SnSclient = new SnsClient([
    'profile' => 'default',
    'region' => 'us-east-1',
    'version' => '2010-03-31'
]);
```

```
try {
    $result = $SnSClient->SetSMSAttributes([
        'attributes' => [
            'DefaultSMSType' => 'Transactional',
        ],
    ]);
    var_dump($result);
} catch (AwsException $e) {
    // output error message if fails
    error_log($e->getMessage());
}
```

- Para obter mais informações, consulte o [Guia do desenvolvedor do AWS SDK for PHP](#).
- Para receber detalhes da API, consulte [SetSMSAttributes](#) na Referência da API AWS SDK for PHP .

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulte [Usando o Amazon SNS com um SDK AWS](#). Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

## Use **SetSubscriptionAttributes** com um AWS SDK ou CLI

Os exemplos de códigos a seguir mostram como usar `SetSubscriptionAttributes`.

### CLI

#### AWS CLI

Para definir atributos de assinatura

O exemplo `set-subscription-attributes` a seguir define o atributo `RawMessageDelivery` para uma assinatura do SQS.

```
aws sns set-subscription-attributes \
    --subscription-arn arn:aws:sns:us-
east-1:123456789012:mytopic:f248de18-2cf6-578c-8592-b6f1eaa877dc \
    --attribute-name RawMessageDelivery \
    --attribute-value true
```

Este comando não produz saída.

O exemplo `set-subscription-attributes` a seguir define um atributo `FilterPolicy` para uma assinatura do SQS.

```
aws sns set-subscription-attributes \  
  --subscription-arn arn:aws:sns:us-  
east-1:123456789012:mytopic:f248de18-2cf6-578c-8592-b6f1eaa877dc \  
  --attribute-name FilterPolicy \  
  --attribute-value "{ \"anyMandatoryKey\": [\"any\", \"of\", \"these\"] }"
```

Este comando não produz saída.

O exemplo `set-subscription-attributes` a seguir remove o atributo `FilterPolicy` de uma assinatura do SQS.

```
aws sns set-subscription-attributes \  
  --subscription-arn arn:aws:sns:us-  
east-1:123456789012:mytopic:f248de18-2cf6-578c-8592-b6f1eaa877dc \  
  --attribute-name FilterPolicy \  
  --attribute-value "{}"
```

Este comando não produz saída.

- Para obter detalhes da API, consulte [SetSubscriptionAttributes](#) na Referência de AWS CLI Comandos.

## Java

### SDK para Java 2.x

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
import software.amazon.awssdk.regions.Region;  
import software.amazon.awssdk.services.sns.SnsClient;  
import software.amazon.awssdk.services.sns.model.SnsException;  
import java.util.ArrayList;
```

```
/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */
public class UseMessageFilterPolicy {
    public static void main(String[] args) {
        final String usage = ""

            Usage:    <subscriptionArn>

            Where:
                subscriptionArn - The ARN of a subscription.

            "";

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String subscriptionArn = args[0];
        SnsClient snsClient = SnsClient.builder()
            .region(Region.US_EAST_1)
            .build();

        usePolicy(snsClient, subscriptionArn);
        snsClient.close();
    }

    public static void usePolicy(SnsClient snsClient, String subscriptionArn) {
        try {
            SNSMessageFilterPolicy fp = new SNSMessageFilterPolicy();
            // Add a filter policy attribute with a single value
            fp.addAttribute("store", "example_corp");
            fp.addAttribute("event", "order_placed");

            // Add a prefix attribute
            fp.addAttributePrefix("customer_interests", "bas");
        }
    }
}
```

```
// Add an anything-but attribute
fp.addAttributeAnythingBut("customer_interests", "baseball");

// Add a filter policy attribute with a list of values
ArrayList<String> attributeValues = new ArrayList<>();
attributeValues.add("rugby");
attributeValues.add("soccer");
attributeValues.add("hockey");
fp.addAttribute("customer_interests", attributeValues);

// Add a numeric attribute
fp.addAttribute("price_usd", "=", 0);

// Add a numeric attribute with a range
fp.addAttributeRange("price_usd", ">", 0, "<=", 100);

// Apply the filter policy attributes to an Amazon SNS subscription
fp.apply(snsClient, subscriptionArn);

} catch (SnsException e) {
    System.err.println(e.awsErrorDetails().errorMessage());
    System.exit(1);
}
}
```

- Para obter detalhes da API, consulte [SetSubscriptionAttributes](#) a Referência AWS SDK for Java 2.x da API.

## Python

### SDK para Python (Boto3)

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
class SnsWrapper:
    """Encapsulates Amazon SNS topic and subscription functions."""

    def __init__(self, sns_resource):
        """
        :param sns_resource: A Boto3 Amazon SNS resource.
        """
        self.sns_resource = sns_resource

    @staticmethod
    def add_subscription_filter(subscription, attributes):
        """
        Adds a filter policy to a subscription. A filter policy is a key and a
        list of values that are allowed. When a message is published, it must
        have an
        attribute that passes the filter or it will not be sent to the
        subscription.

        :param subscription: The subscription the filter policy is attached to.
        :param attributes: A dictionary of key-value pairs that define the
        filter.
        """
        try:
            att_policy = {key: [value] for key, value in attributes.items()}
            subscription.set_attributes(
                AttributeName="FilterPolicy",
                AttributeValue=json.dumps(att_policy)
            )
            logger.info("Added filter to subscription %s.", subscription.arn)
        except ClientError:
            logger.exception(
                "Couldn't add filter to subscription %s.", subscription.arn
            )
            raise
```

- Para obter detalhes da API, consulte a [SetSubscriptionAttributes](#) Referência da API AWS SDK for Python (Boto3).

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulte [Usando o Amazon SNS com um SDK AWS](#). Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

## Use `SetSubscriptionAttributesRedrivePolicy` com um AWS SDK ou CLI

O código de exemplo a seguir mostra como usar `SetSubscriptionAttributesRedrivePolicy`.

Java

SDK para Java 1.x

### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
// Specify the ARN of the Amazon SNS subscription.
String subscriptionArn =
    "arn:aws:sns:us-east-2:123456789012:MyEndpoint:1234a567-
bc89-012d-3e45-6fg7h890123i";

// Specify the ARN of the Amazon SQS queue to use as a dead-letter queue.
String redrivePolicy =
    "{\"deadLetterTargetArn\":\"arn:aws:sqs:us-
east-2:123456789012:MyDeadLetterQueue\"}";

// Set the specified Amazon SQS queue as a dead-letter queue
// of the specified Amazon SNS subscription by setting the RedrivePolicy
// attribute.
SetSubscriptionAttributesRequest request = new SetSubscriptionAttributesRequest()
    .withSubscriptionArn(subscriptionArn)
    .withAttributeName("RedrivePolicy")
    .withAttributeValue(redrivePolicy);
sns.setSubscriptionAttributes(request);
```

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulte [Usando o Amazon SNS com um SDK AWS](#). Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

## Use **SetTopicAttributes** com um AWS SDK ou CLI

Os exemplos de códigos a seguir mostram como usar `SetTopicAttributes`.

### CLI

#### AWS CLI

Para definir um atributo para um tópico

O exemplo `set-topic-attributes` a seguir define o atributo `DisplayName` para o tópico especificado.

```
aws sns set-topic-attributes \  
  --topic-arn arn:aws:sns:us-west-2:123456789012:MyTopic \  
  --attribute-name DisplayName \  
  --attribute-value MyTopicDisplayName
```

Este comando não produz saída.

- Para obter detalhes da API, consulte [SetTopicAttributes](#) na Referência de AWS CLI Comandos.

### Java

#### SDK para Java 2.x

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
import software.amazon.awssdk.regions.Region;  
import software.amazon.awssdk.services.sns.SnsClient;  
import software.amazon.awssdk.services.sns.model.SetTopicAttributesRequest;  
import software.amazon.awssdk.services.sns.model.SetTopicAttributesResponse;
```

```
import software.amazon.awssdk.services.sns.model.SnsException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */
public class SetTopicAttributes {

    public static void main(String[] args) {
        final String usage = ""

            Usage:    <attribute> <topicArn> <value>

            Where:
                attribute - The attribute action to use. Valid parameters are:
Policy | DisplayName | DeliveryPolicy .
                topicArn - The ARN of the topic.\s
                value - The value for the attribute.
            """;

        if (args.length < 3) {
            System.out.println(usage);
            System.exit(1);
        }

        String attribute = args[0];
        String topicArn = args[1];
        String value = args[2];

        SnsClient snsClient = SnsClient.builder()
            .region(Region.US_EAST_1)
            .build();

        setTopAttr(snsClient, attribute, topicArn, value);
        snsClient.close();
    }

    public static void setTopAttr(SnsClient snsClient, String attribute, String
topicArn, String value) {
```

```
    try {
        SetTopicAttributesRequest request =
SetTopicAttributesRequest.builder()
        .attributeName(attribute)
        .attributeValue(value)
        .topicArn(topicArn)
        .build();

        SetTopicAttributesResponse result =
snsClient.setTopicAttributes(request);
        System.out.println(
            "\n\nStatus was " + result.sdkHttpResponse().statusCode() +
"\n\nTopic " + request.topicArn()
            + " updated " + request.attributeName() + " to " +
request.attributeValue());

    } catch (SnsException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}
}
```

- Para obter detalhes da API, consulte [SetTopicAttributes](#) na Referência AWS SDK for Java 2.x da API.

## JavaScript

### SDK para JavaScript (v3)

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

Crie o cliente em um módulo separado e exporte-o.

```
import { SNSClient } from "@aws-sdk/client-sns";
```

```
// The AWS Region can be provided here using the `region` property. If you leave
it blank
// the SDK will default to the region set in your AWS config.
export const snsClient = new SNSClient({});
```

Importe o SDK e os módulos do cliente e chame a API.

```
import { SetTopicAttributesCommand } from "@aws-sdk/client-sns";
import { snsClient } from "../libs/snsClient.js";

export const setTopicAttributes = async (
  topicArn = "TOPIC_ARN",
  attributeName = "DisplayName",
  attributeValue = "Test Topic",
) => {
  const response = await snsClient.send(
    new SetTopicAttributesCommand({
      AttributeName: attributeName,
      AttributeValue: attributeValue,
      TopicArn: topicArn,
    }),
  );
  console.log(response);
  // {
  //   '$metadata': {
  //     httpStatusCode: 200,
  //     requestId: 'd1b08d0e-e9a4-54c3-b8b1-d03238d2b935',
  //     extendedRequestId: undefined,
  //     cfId: undefined,
  //     attempts: 1,
  //     totalRetryDelay: 0
  //   }
  // }
  return response;
};
```

- Para obter mais informações, consulte o [Guia do desenvolvedor do AWS SDK for JavaScript](#).
- Para obter detalhes da API, consulte [SetTopicAttributes](#) a Referência AWS SDK for JavaScript da API.

## Kotlin

### SDK para Kotlin

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
suspend fun setTopAttr(attribute: String?, topicArnVal: String?, value: String?)
{
    val request = SetTopicAttributesRequest {
        attributeName = attribute
        attributeValue = value
        topicArn = topicArnVal
    }

    SnsClient { region = "us-east-1" }.use { snsClient ->
        snsClient.setTopicAttributes(request)
        println("Topic ${request.topicArn} was updated.")
    }
}
```

- Para obter detalhes da API, consulte a [SetTopicAttributes](#) referência da API AWS SDK for Kotlin.

## PHP

### SDK para PHP

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
require 'vendor/autoload.php';
```

```
use Aws\Exception\AwsException;
use Aws\Sns\SnsClient;

/**
 * Configure the message delivery status attributes for an Amazon SNS Topic.
 *
 * This code expects that you have AWS credentials set up per:
 * https://docs.aws.amazon.com/sdk-for-php/v3/developer-guide/
guide_credentials.html
 */

$SnSClient = new SnsClient([
    'profile' => 'default',
    'region' => 'us-east-1',
    'version' => '2010-03-31'
]);
$attribute = 'Policy | DisplayName | DeliveryPolicy';
$value = 'First Topic';
$topic = 'arn:aws:sns:us-east-1:111122223333:MyTopic';

try {
    $result = $SnSClient->setTopicAttributes([
        'AttributeName' => $attribute,
        'AttributeValue' => $value,
        'TopicArn' => $topic,
    ]);
    var_dump($result);
} catch (AwsException $e) {
    // output error message if fails
    error_log($e->getMessage());
}
```

- Para obter detalhes da API, consulte [SetTopicAttributes](#) na Referência AWS SDK for PHP da API.

## Ruby

### SDK para Ruby

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
# Service class to enable an SNS resource with a specified policy
class SnsResourceEnabler
  # Initializes the SnsResourceEnabler with an SNS resource client
  #
  # @param sns_resource [Aws::SNS::Resource] The SNS resource client
  def initialize(sns_resource)
    @sns_resource = sns_resource
    @logger = Logger.new($stdout)
  end

  # Sets a policy on a specified SNS topic
  #
  # @param topic_arn [String] The ARN of the SNS topic
  # @param resource_arn [String] The ARN of the resource to include in the policy
  # @param policy_name [String] The name of the policy attribute to set
  def enable_resource(topic_arn, resource_arn, policy_name)
    policy = generate_policy(topic_arn, resource_arn)
    topic = @sns_resource.topic(topic_arn)

    topic.set_attributes({
      attribute_name: policy_name,
      attribute_value: policy
    })

    @logger.info("Policy #{policy_name} set successfully for topic
#{topic_arn}.")
    rescue Aws::SNS::Errors::ServiceError => e
      @logger.error("Failed to set policy: #{e.message}")
    end

  private

  # Generates a policy string with dynamic resource ARNs
```

```
#
# @param topic_arn [String] The ARN of the SNS topic
# @param resource_arn [String] The ARN of the resource
# @return [String] The policy as a JSON string
def generate_policy(topic_arn, resource_arn)
  {
    Version: "2008-10-17",
    Id: "__default_policy_ID",
    Statement: [{
      Sid: "__default_statement_ID",
      Effect: "Allow",
      Principal: { "AWS": "*" },
      Action: ["SNS:Publish"],
      Resource: topic_arn,
      Condition: {
        ArnEquals: {
          "AWS:SourceArn": resource_arn
        }
      }
    }]
  }.to_json
end
end

# Example usage:
if $PROGRAM_NAME == __FILE__
  topic_arn = "MY_TOPIC_ARN"      # Should be replaced with a real topic ARN
  resource_arn = "MY_RESOURCE_ARN" # Should be replaced with a real resource ARN
  policy_name = "POLICY_NAME"    # Typically, this is "Policy"

  sns_resource = Aws::SNS::Resource.new
  enabler = SnsResourceEnabler.new(sns_resource)

  enabler.enable_resource(topic_arn, resource_arn, policy_name)
end
```

- Para obter mais informações, consulte o [Guia do desenvolvedor do AWS SDK for Ruby](#).
- Para obter detalhes da API, consulte [SetTopicAttributes](#) a Referência AWS SDK for Ruby da API.

## SAP ABAP

### SDK para SAP ABAP

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
TRY.  
    lo_sns->settopicattributes(  
        iv_topicarn = iv_topic_arn  
        iv_attributename = iv_attribute_name  
        iv_attributevalue = iv_attribute_value  
    ).  
    MESSAGE 'Set/updated SNS topic attributes.' TYPE 'I'.  
CATCH /aws1/cx_snsnotfoundexception.  
    MESSAGE 'Topic does not exist.' TYPE 'E'.  
ENDTRY.
```

- Para obter detalhes da API, consulte a [SetTopicAttributes](#) referência da API AWS SDK for SAP ABAP.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulte [Usando o Amazon SNS com um SDK AWS](#). Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

## Use **Subscribe** com um AWS SDK ou CLI

Os exemplos de códigos a seguir mostram como usar `Subscribe`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto nos seguintes exemplos de código:

- [Criar e publicar em um tópico FIFO](#)
- [Publicar mensagens em filas](#)

## .NET

### AWS SDK for .NET

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

Inscreva um endereço de e-mail em um tópico.

```
/// <summary>
/// Creates a new subscription to a topic.
/// </summary>
/// <param name="client">The initialized Amazon SNS client object, used
/// to create an Amazon SNS subscription.</param>
/// <param name="topicArn">The ARN of the topic to subscribe to.</param>
/// <returns>A SubscribeResponse object which includes the subscription
/// ARN for the new subscription.</returns>
public static async Task<SubscribeResponse> TopicSubscribeAsync(
    IAmazonSimpleNotificationService client,
    string topicArn)
{
    SubscribeRequest request = new SubscribeRequest()
    {
        TopicArn = topicArn,
        ReturnSubscriptionArn = true,
        Protocol = "email",
        Endpoint = "recipient@example.com",
    };

    var response = await client.SubscribeAsync(request);

    return response;
}
```

Inscreva uma fila em um tópico com filtros opcionais.

```
/// <summary>
/// Subscribe a queue to a topic with optional filters.
/// </summary>
/// <param name="topicArn">The ARN of the topic.</param>
/// <param name="useFifoTopic">The optional filtering policy for the
subscription.</param>
/// <param name="queueArn">The ARN of the queue.</param>
/// <returns>The ARN of the new subscription.</returns>
public async Task<string> SubscribeTopicWithFilter(string topicArn, string?
filterPolicy, string queueArn)
{
    var subscribeRequest = new SubscribeRequest()
    {
        TopicArn = topicArn,
        Protocol = "sqs",
        Endpoint = queueArn
    };

    if (!string.IsNullOrEmpty(filterPolicy))
    {
        subscribeRequest.Attributes = new Dictionary<string, string>
{ { "FilterPolicy", filterPolicy } };
    }

    var subscribeResponse = await
_amazonSNSClient.SubscribeAsync(subscribeRequest);
    return subscribeResponse.SubscriptionArn;
}
```

- Para obter detalhes da API, consulte [Subscribe](#) na Referência da API AWS SDK for .NET .

## C++

### SDK para C++

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

## Inscreva um endereço de e-mail em um tópico.

```
#!/ Subscribe to an Amazon Simple Notification Service (Amazon SNS) topic with
delivery to an email address.
/*!
 \param topicARN: An SNS topic Amazon Resource Name (ARN).
 \param emailAddress: An email address.
 \param clientConfiguration: AWS client configuration.
 \return bool: Function succeeded.
 */
bool AwsDoc::SNS::subscribeEmail(const Aws::String &topicARN,
                                const Aws::String &emailAddress,
                                const Aws::Client::ClientConfiguration
&clientConfiguration) {
    Aws::SNS::SNSClient snsClient(clientConfiguration);

    Aws::SNS::Model::SubscribeRequest request;
    request.SetTopicArn(topicARN);
    request.SetProtocol("email");
    request.SetEndpoint(emailAddress);

    const Aws::SNS::Model::SubscribeOutcome outcome =
snsClient.Subscribe(request);

    if (outcome.IsSuccess()) {
        std::cout << "Subscribed successfully." << std::endl;
        std::cout << "Subscription ARN " <<
outcome.GetResult().GetSubscriptionArn()
        << "." << std::endl;
    }
    else {
        std::cerr << "Error while subscribing " <<
outcome.GetError().GetMessage()
        << std::endl;
    }

    return outcome.IsSuccess();
}
```

## Inscreva um aplicativo móvel em um tópico.

```

//! Subscribe to an Amazon Simple Notification Service (Amazon SNS) topic with
delivery to a mobile app.
/*!
  \param topicARN: The Amazon Resource Name (ARN) for an Amazon SNS topic.
  \param endpointARN: The ARN for a mobile app or device endpoint.
  \param clientConfiguration: AWS client configuration.
  \return bool: Function succeeded.
*/
bool
AwsDoc::SNS::subscribeApp(const Aws::String &topicARN,
                          const Aws::String &endpointARN,
                          const Aws::Client::ClientConfiguration
&clientConfiguration) {
    Aws::SNS::SNSClient snsClient(clientConfiguration);

    Aws::SNS::Model::SubscribeRequest request;
    request.SetTopicArn(topicARN);
    request.SetProtocol("application");
    request.SetEndpoint(endpointARN);

    const Aws::SNS::Model::SubscribeOutcome outcome =
snsClient.Subscribe(request);

    if (outcome.IsSuccess()) {
        std::cout << "Subscribed successfully." << std::endl;
        std::cout << "Subscription ARN '" <<
outcome.GetResult().GetSubscriptionArn()
        << "'." << std::endl;
    }
    else {
        std::cerr << "Error while subscribing " <<
outcome.GetError().GetMessage()
        << std::endl;
    }

    return outcome.IsSuccess();
}

```

Inscreva uma função Lambda em um tópico.

```

//! Subscribe to an Amazon Simple Notification Service (Amazon SNS) topic with
delivery to an AWS Lambda function.
/*!
  \param topicARN: The Amazon Resource Name (ARN) for an Amazon SNS topic.
  \param lambdaFunctionARN: The ARN for an AWS Lambda function.
  \param clientConfiguration: AWS client configuration.
  \return bool: Function succeeded.
*/
bool AwsDoc::SNS::subscribeLambda(const Aws::String &topicARN,
                                  const Aws::String &lambdaFunctionARN,
                                  const Aws::Client::ClientConfiguration
&clientConfiguration) {
    Aws::SNS::SNSClient snsClient(clientConfiguration);

    Aws::SNS::Model::SubscribeRequest request;
    request.SetTopicArn(topicARN);
    request.SetProtocol("lambda");
    request.SetEndpoint(lambdaFunctionARN);

    const Aws::SNS::Model::SubscribeOutcome outcome =
snsClient.Subscribe(request);

    if (outcome.IsSuccess()) {
        std::cout << "Subscribed successfully." << std::endl;
        std::cout << "Subscription ARN '" <<
outcome.GetResult().GetSubscriptionArn()
        << "'." << std::endl;
    }
    else {
        std::cerr << "Error while subscribing " <<
outcome.GetError().GetMessage()
        << std::endl;
    }

    return outcome.IsSuccess();
}

```

Inscreva uma fila SQS em um tópico.

```

Aws::Client::ClientConfiguration clientConfig;
// Optional: Set to the AWS Region (overrides config file).
// clientConfig.region = "us-east-1";

```

```

    Aws::SNS::SNSClient snsClient(clientConfiguration);

    Aws::SNS::Model::SubscribeRequest request;
    request.SetTopicArn(topicARN);
    request.SetProtocol("sqs");
    request.SetEndpoint(queueARN);

    Aws::SNS::Model::SubscribeOutcome outcome =
snsClient.Subscribe(request);

    if (outcome.IsSuccess()) {
        Aws::String subscriptionARN =
outcome.GetResult().GetSubscriptionArn();
        std::cout << "The queue '" << queueName
            << "' has been subscribed to the topic '"
            << "'" << topicName << "'" << std::endl;
        std::cout << "with the subscription ARN '" << subscriptionARN <<
". "
            << std::endl;
        subscriptionARNS.push_back(subscriptionARN);
    }
    else {
        std::cerr << "Error with TopicsAndQueues::Subscribe. "
            << outcome.GetError().GetMessage()
            << std::endl;

        cleanUp(topicARN,
            queueURLS,
            subscriptionARNS,
            snsClient,
            sqsClient);

        return false;
    }
}

```

Inscreva-se com um filtro para um tópico.

```

static const Aws::String TONE_ATTRIBUTE("tone");
static const Aws::Vector<Aws::String> TONES = {"cheerful", "funny",
"serious",
"sincere"};

```

```

    Aws::Client::ClientConfiguration clientConfig;
    // Optional: Set to the AWS Region (overrides config file).
    // clientConfig.region = "us-east-1";

    Aws::SNS::SNSClient snsClient(clientConfiguration);

    Aws::SNS::Model::SubscribeRequest request;
    request.SetTopicArn(topicARN);
    request.SetProtocol("sqs");
    request.SetEndpoint(queueARN);
    if (isFifoTopic) {
        if (first) {
            std::cout << "Subscriptions to a FIFO topic can have
filters."
                        << std::endl;
            std::cout
                << "If you add a filter to this subscription, then
only the filtered messages "
                << "will be received in the queue." << std::endl;
            std::cout << "For information about message filtering, "
                << "see https://docs.aws.amazon.com/sns/latest/dg/
sns-message-filtering.html"
                << std::endl;
            std::cout << "For this example, you can filter messages by a
\""
                << TONE_ATTRIBUTE << "\" attribute." << std::endl;
        }

        std::ostringstream ostream;
        ostream << "Filter messages for \"" << queueName
                << "\"'s subscription to the topic \""
                << topicName << "\"? (y/n)";

        // Add filter if user answers yes.
        if (askYesNoQuestion(ostream.str())) {
            Aws::String jsonPolicy = getFilterPolicyFromUser();
            if (!jsonPolicy.empty()) {
                filteringMessages = true;

                std::cout << "This is the filter policy for this
subscription."
                        << std::endl;
                std::cout << jsonPolicy << std::endl;
            }
        }
    }
}

```

```

        request.AddAttributes("FilterPolicy", jsonPolicy);
    }
    else {
        std::cout
            << "Because you did not select any attributes, no
filter "
            << "will be added to this subscription." <<
std::endl;
    }
}
} // if (isFifoTopic)
Aws::SNS::Model::SubscribeOutcome outcome =
snsClient.Subscribe(request);

if (outcome.IsSuccess()) {
    Aws::String subscriptionARN =
outcome.GetResult().GetSubscriptionArn();
    std::cout << "The queue '" << queueName
        << "' has been subscribed to the topic '"
        << "'" << topicName << "'" << std::endl;
    std::cout << "with the subscription ARN '" << subscriptionARN <<
"."
        << std::endl;
    subscriptionARNS.push_back(subscriptionARN);
}
else {
    std::cerr << "Error with TopicsAndQueues::Subscribe. "
        << outcome.GetError().GetMessage()
        << std::endl;

    cleanUp(topicARN,
        queueURLS,
        subscriptionARNS,
        snsClient,
        sqsClient);

    return false;
}

//! Routine that lets the user select attributes for a subscription filter
policy.
/*!
\sa getFilterPolicyFromUser()

```

```
\return Aws::String: The filter policy as JSON.
*/
Aws::String AwsDoc::TopicsAndQueues::getFilterPolicyFromUser() {
    std::cout
        << "You can filter messages by one or more of the following \""
        << TONE_ATTRIBUTE << "\" attributes." << std::endl;

    std::vector<Aws::String> filterSelections;
    int selection;
    do {
        for (size_t j = 0; j < TONES.size(); ++j) {
            std::cout << " " << (j + 1) << ". " << TONES[j]
                << std::endl;
        }
        selection = askQuestionForIntRange(
            "Enter a number (or enter zero to stop adding more). ",
            0, static_cast<int>(TONES.size()));

        if (selection != 0) {
            const Aws::String &selectedTone(TONES[selection - 1]);
            // Add the tone to the selection if it is not already added.
            if (std::find(filterSelections.begin(),
                filterSelections.end(),
                selectedTone)
                == filterSelections.end()) {
                filterSelections.push_back(selectedTone);
            }
        }
    } while (selection != 0);

    Aws::String result;
    if (!filterSelections.empty()) {
        std::ostringstream jsonPolicyStream;
        jsonPolicyStream << "{ \"" << TONE_ATTRIBUTE << "\": [";

        for (size_t j = 0; j < filterSelections.size(); ++j) {
            jsonPolicyStream << "\"" << filterSelections[j] << "\"";
            if (j < filterSelections.size() - 1) {
                jsonPolicyStream << ",";
            }
        }
        jsonPolicyStream << "] }";
    }
}
```

```
        result = jsonPolicyStream.str();
    }

    return result;
}
```

- Para obter detalhes da API, consulte [Subscribe](#) na Referência da API AWS SDK for C++ .

## CLI

### AWS CLI

Para inscrever-se em um tópico

O comando `subscribe` a seguir inscreve um endereço de e-mail no tópico especificado.

```
aws sns subscribe \
  --topic-arn arn:aws:sns:us-west-2:123456789012:my-topic \
  --protocol email \
  --notification-endpoint my-email@example.com
```

Saída:

```
{
  "SubscriptionArn": "pending confirmation"
}
```

- Para obter detalhes da API, consulte [Subscribe](#) na Referência de comandos da AWS CLI .

## Go

### SDK para Go V2

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

Inscreva uma fila em um tópico com filtros opcionais.

```
// SnsActions encapsulates the Amazon Simple Notification Service (Amazon SNS)
actions
// used in the examples.
type SnsActions struct {
    SnsClient *sns.Client
}

// SubscribeQueue subscribes an Amazon Simple Queue Service (Amazon SQS) queue to
an
// Amazon SNS topic. When filterMap is not nil, it is used to specify a filter
policy
// so that messages are only sent to the queue when the message has the specified
attributes.
func (actor SnsActions) SubscribeQueue(topicArn string, queueArn string,
filterMap map[string][]string) (string, error) {
    var subscriptionArn string
    var attributes map[string]string
    if filterMap != nil {
        filterBytes, err := json.Marshal(filterMap)
        if err != nil {
            log.Printf("Couldn't create filter policy, here's why: %v\n", err)
            return "", err
        }
        attributes = map[string]string{"FilterPolicy": string(filterBytes)}
    }
    output, err := actor.SnsClient.Subscribe(context.TODO(), &sns.SubscribeInput{
        Protocol:          aws.String("sqs"),
        TopicArn:          aws.String(topicArn),
        Attributes:        attributes,
        Endpoint:          aws.String(queueArn),
        ReturnSubscriptionArn: true,
    })
    if err != nil {
        log.Printf("Couldn't subscribe queue %v to topic %v. Here's why: %v\n",
            queueArn, topicArn, err)
    } else {
        subscriptionArn = *output.SubscriptionArn
    }

    return subscriptionArn, err
}
```

```
}
```

- Para obter detalhes da API, consulte [Subscribe](#) na Referência da API AWS SDK for Go .

## Java

### SDK para Java 2.x

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

Inscreva um endereço de e-mail em um tópico.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.sns.SnsClient;
import software.amazon.awssdk.services.sns.model.SnsException;
import software.amazon.awssdk.services.sns.model.SubscribeRequest;
import software.amazon.awssdk.services.sns.model.SubscribeResponse;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
 * started.html
 */
public class SubscribeEmail {
    public static void main(String[] args) {
        final String usage = ""
            Usage:    <topicArn> <email>

            Where:
                topicArn - The ARN of the topic to subscribe.
                email - The email address to use.
            """;
    }
}
```

```
        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String topicArn = args[0];
        String email = args[1];
        SnsClient snsClient = SnsClient.builder()
            .region(Region.US_EAST_1)
            .build();

        subEmail(snsClient, topicArn, email);
        snsClient.close();
    }

    public static void subEmail(SnsClient snsClient, String topicArn, String
email) {
        try {
            SubscribeRequest request = SubscribeRequest.builder()
                .protocol("email")
                .endpoint(email)
                .returnSubscriptionArn(true)
                .topicArn(topicArn)
                .build();

            SubscribeResponse result = snsClient.subscribe(request);
            System.out.println("Subscription ARN: " + result.subscriptionArn() +
"\n\n Status is "
                + result.sdkHttpResponse().statusCode());

        } catch (SnsException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

Inscreva um endpoint HTTP em um tópico.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.sns.SnsClient;
```

```
import software.amazon.awssdk.services.sns.model.SnsException;
import software.amazon.awssdk.services.sns.model.SubscribeRequest;
import software.amazon.awssdk.services.sns.model.SubscribeResponse;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */
public class SubscribeHTTPS {
    public static void main(String[] args) {
        final String usage = ""

            Usage:    <topicArn> <url>

            Where:
                topicArn - The ARN of the topic to subscribe.
                url - The HTTPS endpoint that you want to receive
notifications.
            """;

        if (args.length < 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String topicArn = args[0];
        String url = args[1];
        SnsClient snsClient = SnsClient.builder()
            .region(Region.US_EAST_1)
            .build();

        subHTTPS(snsClient, topicArn, url);
        snsClient.close();
    }

    public static void subHTTPS(SnsClient snsClient, String topicArn, String url)
    {
        try {
            SubscribeRequest request = SubscribeRequest.builder()
```

```

        .protocol("https")
        .endpoint(url)
        .returnSubscriptionArn(true)
        .topicArn(topicArn)
        .build();

        SubscribeResponse result = snsClient.subscribe(request);
        System.out.println("Subscription ARN is " + result.subscriptionArn()
+ "\n\n Status is "
        + result.sdkHttpResponse().statusCode());

    } catch (SnsException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}
}
}

```

Inscreva uma função Lambda em um tópico.

```

import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.sns.SnsClient;
import software.amazon.awssdk.services.sns.model.SnsException;
import software.amazon.awssdk.services.sns.model.SubscribeRequest;
import software.amazon.awssdk.services.sns.model.SubscribeResponse;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */
public class SubscribeLambda {

    public static void main(String[] args) {

        final String usage = ""

            Usage:    <topicArn> <lambdaArn>

```

```
        Where:
            topicArn - The ARN of the topic to subscribe.
            lambdaArn - The ARN of an AWS Lambda function.
        """;

    if (args.length != 2) {
        System.out.println(usage);
        System.exit(1);
    }

    String topicArn = args[0];
    String lambdaArn = args[1];
    SnsClient snsClient = SnsClient.builder()
        .region(Region.US_EAST_1)
        .build();

    String arnValue = subLambda(snsClient, topicArn, lambdaArn);
    System.out.println("Subscription ARN: " + arnValue);
    snsClient.close();
}

public static String subLambda(SnsClient snsClient, String topicArn, String
lambdaArn) {
    try {
        SubscribeRequest request = SubscribeRequest.builder()
            .protocol("lambda")
            .endpoint(lambdaArn)
            .returnSubscriptionArn(true)
            .topicArn(topicArn)
            .build();

        SubscribeResponse result = snsClient.subscribe(request);
        return result.subscriptionArn();

    } catch (SnsException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
    return "";
}
}
```

- Para obter detalhes da API, consulte [Subscribe](#) na Referência da API AWS SDK for Java 2.x .

## JavaScript

### SDK para JavaScript (v3)

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

Crie o cliente em um módulo separado e exporte-o.

```
import { SNSClient } from "@aws-sdk/client-sns";

// The AWS Region can be provided here using the `region` property. If you leave
// it blank
// the SDK will default to the region set in your AWS config.
export const snsClient = new SNSClient({});
```

Importe o SDK e os módulos do cliente e chame a API.

```
import { SubscribeCommand } from "@aws-sdk/client-sns";
import { snsClient } from "../libs/snsClient.js";

/**
 * @param {string} topicArn - The ARN of the topic for which you wish to confirm
 * a subscription.
 * @param {string} emailAddress - The email address that is subscribed to the
 * topic.
 */
export const subscribeEmail = async (
  topicArn = "TOPIC_ARN",
  emailAddress = "user@me.com",
) => {
  const response = await snsClient.send(
    new SubscribeCommand({
```

```

        Protocol: "email",
        TopicArn: topicArn,
        Endpoint: emailAddress,
    })),
);
console.log(response);
// {
//   '$metadata': {
//     httpStatusCode: 200,
//     requestId: 'c8e35bcd-b3c0-5940-9f66-06f6fcc108f0',
//     extendedRequestId: undefined,
//     cfId: undefined,
//     attempts: 1,
//     totalRetryDelay: 0
//   },
//   SubscriptionArn: 'pending confirmation'
// }
};

```

Inscreva um aplicativo móvel em um tópico.

```

import { SubscribeCommand } from "@aws-sdk/client-sns";
import { snsClient } from "../libs/snsClient.js";

/**
 * @param {string} topicArn - The ARN of the topic the subscriber is subscribing
 * to.
 * @param {string} endpoint - The Endpoint ARN of an application. This endpoint
 * is created
 *
 *                               when an application registers for notifications.
 */
export const subscribeApp = async (
  topicArn = "TOPIC_ARN",
  endpoint = "ENDPOINT",
) => {
  const response = await snsClient.send(
    new SubscribeCommand({
      Protocol: "application",
      TopicArn: topicArn,
      Endpoint: endpoint,
    }),
  );
};

```

```
console.log(response);
// {
//   '$metadata': {
//     httpStatusCode: 200,
//     requestId: 'c8e35bcd-b3c0-5940-9f66-06f6fcc108f0',
//     extendedRequestId: undefined,
//     cfId: undefined,
//     attempts: 1,
//     totalRetryDelay: 0
//   },
//   SubscriptionArn: 'pending confirmation'
// }
return response;
};
```

Inscreva uma função Lambda em um tópico.

```
import { SubscribeCommand } from "@aws-sdk/client-sns";
import { snsClient } from "../libs/snsClient.js";

/**
 * @param {string} topicArn - The ARN of the topic the subscriber is subscribing
 * to.
 * @param {string} endpoint - The Endpoint ARN of and AWS Lambda function.
 */
export const subscribeLambda = async (
  topicArn = "TOPIC_ARN",
  endpoint = "ENDPOINT",
) => {
  const response = await snsClient.send(
    new SubscribeCommand({
      Protocol: "lambda",
      TopicArn: topicArn,
      Endpoint: endpoint,
    }),
  );
  console.log(response);
  // {
  //   '$metadata': {
  //     httpStatusCode: 200,
  //     requestId: 'c8e35bcd-b3c0-5940-9f66-06f6fcc108f0',
  //     extendedRequestId: undefined,
```

```
//    cfId: undefined,  
//    attempts: 1,  
//    totalRetryDelay: 0  
//  },  
//  SubscriptionArn: 'pending confirmation'  
// }  
return response;  
};
```

Inscreva uma fila SQS em um tópico.

```
import { SubscribeCommand, SNSClient } from "@aws-sdk/client-sns";  
  
const client = new SNSClient({});  
  
export const subscribeQueue = async (  
  topicArn = "TOPIC_ARN",  
  queueArn = "QUEUE_ARN",  
) => {  
  const command = new SubscribeCommand({  
    TopicArn: topicArn,  
    Protocol: "sqs",  
    Endpoint: queueArn,  
  });  
  
  const response = await client.send(command);  
  console.log(response);  
  // {  
  //   '$metadata': {  
  //     httpStatusCode: 200,  
  //     requestId: '931e13d9-5e2b-543f-8781-4e9e494c5ff2',  
  //     extendedRequestId: undefined,  
  //     cfId: undefined,  
  //     attempts: 1,  
  //     totalRetryDelay: 0  
  //   },  
  //   SubscriptionArn: 'arn:aws:sns:us-east-1:xxxxxxxxxxxx:subscribe-queue-  
test-430895:xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx'  
  // }  
  return response;  
};
```

## Inscreva-se com um filtro para um tópico.

```
import { SubscribeCommand, SNSClient } from "@aws-sdk/client-sns";

const client = new SNSClient({});

export const subscribeQueueFiltered = async (
  topicArn = "TOPIC_ARN",
  queueArn = "QUEUE_ARN",
) => {
  const command = new SubscribeCommand({
    TopicArn: topicArn,
    Protocol: "sqs",
    Endpoint: queueArn,
    Attributes: {
      // This subscription will only receive messages with the 'event' attribute
      // set to 'order_placed'.
      FilterPolicyScope: "MessageAttributes",
      FilterPolicy: JSON.stringify({
        event: ["order_placed"],
      }),
    },
  });

  const response = await client.send(command);
  console.log(response);
  // {
  //   '$metadata': {
  //     httpStatusCode: 200,
  //     requestId: '931e13d9-5e2b-543f-8781-4e9e494c5ff2',
  //     extendedRequestId: undefined,
  //     cfId: undefined,
  //     attempts: 1,
  //     totalRetryDelay: 0
  //   },
  //   SubscriptionArn: 'arn:aws:sns:us-east-1:xxxxxxxxxxxx:subscribe-queue-
  // test-430895:xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxxx'
  // }
  return response;
};
```

- Para obter mais informações, consulte o [Guia do desenvolvedor do AWS SDK for JavaScript](#).
- Para obter detalhes da API, consulte [Subscribe](#) na Referência da API AWS SDK for JavaScript .

## Kotlin

### SDK para Kotlin

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

Inscreva um endereço de e-mail em um tópico.

```
suspend fun subEmail(topicArnVal: String, email: String): String {  
  
    val request = SubscribeRequest {  
        protocol = "email"  
        endpoint = email  
        returnSubscriptionArn = true  
        topicArn = topicArnVal  
    }  
  
    SnsClient { region = "us-east-1" }.use { snsClient ->  
        val result = snsClient.subscribe(request)  
        return result.subscriptionArn.toString()  
    }  
}
```

Inscreva uma função Lambda em um tópico.

```
suspend fun subLambda(topicArnVal: String?, lambdaArn: String?) {  
  
    val request = SubscribeRequest {  
        protocol = "lambda"  
        endpoint = lambdaArn  
    }  
}
```

```
        returnSubscriptionArn = true
        topicArn = topicArnVal
    }

    SnsClient { region = "us-east-1" }.use { snsClient ->
        val result = snsClient.subscribe(request)
        println(" The subscription Arn is ${result.subscriptionArn}")
    }
}
```

- Para obter detalhes da API, consulte [Subscribe](#) na Referência da API AWS SDK para Kotlin.

## PHP

### SDK para PHP

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

Inscreva um endereço de e-mail em um tópico.

```
require 'vendor/autoload.php';

use Aws\Exception\AwsException;
use Aws\Sns\SnsClient;

/**
 * Prepares to subscribe an endpoint by sending the endpoint a confirmation
 * message.
 *
 * This code expects that you have AWS credentials set up per:
 * https://docs.aws.amazon.com/sdk-for-php/v3/developer-guide/
 * guide_credentials.html
 */

$SnsClient = new SnsClient([
    'profile' => 'default',
```

```
'region' => 'us-east-1',
'version' => '2010-03-31'
]);

$protocol = 'email';
$endpoint = 'sample@example.com';
$topic = 'arn:aws:sns:us-east-1:111122223333:MyTopic';

try {
    $result = $SnSClient->subscribe([
        'Protocol' => $protocol,
        'Endpoint' => $endpoint,
        'ReturnSubscriptionArn' => true,
        'TopicArn' => $topic,
    ]);
    var_dump($result);
} catch (AwsException $e) {
    // output error message if fails
    error_log($e->getMessage());
}
```

Inscreva um endpoint HTTP em um tópico.

```
require 'vendor/autoload.php';

use Aws\Exception\AwsException;
use Aws\Sns\SnsClient;

/**
 * Prepares to subscribe an endpoint by sending the endpoint a confirmation
 * message.
 *
 * This code expects that you have AWS credentials set up per:
 * https://docs.aws.amazon.com/sdk-for-php/v3/developer-guide/
 * guide_credentials.html
 */

$SnSClient = new SnsClient([
    'profile' => 'default',
    'region' => 'us-east-1',
```

```
'version' => '2010-03-31'
]);

$protocol = 'https';
$endpoint = 'https://';
$topic = 'arn:aws:sns:us-east-1:111122223333:MyTopic';

try {
    $result = $SnSClient->subscribe([
        'Protocol' => $protocol,
        'Endpoint' => $endpoint,
        'ReturnSubscriptionArn' => true,
        'TopicArn' => $topic,
    ]);
    var_dump($result);
} catch (AwsException $e) {
    // output error message if fails
    error_log($e->getMessage());
}
```

- Para obter detalhes da API, consulte [Subscribe](#) na Referência da API AWS SDK for PHP .

## Python

### SDK para Python (Boto3)

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

Inscreva um endereço de e-mail em um tópico.

```
class SnsWrapper:
    """Encapsulates Amazon SNS topic and subscription functions."""

    def __init__(self, sns_resource):
        """
        :param sns_resource: A Boto3 Amazon SNS resource.
```

```
    """
    self.sns_resource = sns_resource

    @staticmethod
    def subscribe(topic, protocol, endpoint):
        """
        Subscribes an endpoint to the topic. Some endpoint types, such as email,
        must be confirmed before their subscriptions are active. When a
        subscription
        is not confirmed, its Amazon Resource Number (ARN) is set to
        'PendingConfirmation'.

        :param topic: The topic to subscribe to.
        :param protocol: The protocol of the endpoint, such as 'sms' or 'email'.
        :param endpoint: The endpoint that receives messages, such as a phone
        number
                        (in E.164 format) for SMS messages, or an email address
        for
                        email messages.
        :return: The newly added subscription.
        """
        try:
            subscription = topic.subscribe(
                Protocol=protocol, Endpoint=endpoint, ReturnSubscriptionArn=True
            )
            logger.info("Subscribed %s %s to topic %s.", protocol, endpoint,
                topic.arn)
        except ClientError:
            logger.exception(
                "Couldn't subscribe %s %s to topic %s.", protocol, endpoint,
                topic.arn
            )
            raise
        else:
            return subscription
```

- Para obter detalhes da API, consulte [Subscribe](#) na Referência da API AWS SDK for Python (Boto3).

## Ruby

### SDK para Ruby

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

Inscreva um endereço de e-mail em um tópico.

```
require "aws-sdk-sns"
require "logger"

# Represents a service for creating subscriptions in Amazon Simple Notification
# Service (SNS)
class SubscriptionService
  # Initializes the SubscriptionService with an SNS client
  #
  # @param sns_client [Aws::SNS::Client] The SNS client
  def initialize(sns_client)
    @sns_client = sns_client
    @logger = Logger.new($stdout)
  end

  # Attempts to create a subscription to a topic
  #
  # @param topic_arn [String] The ARN of the SNS topic
  # @param protocol [String] The subscription protocol (e.g., email)
  # @param endpoint [String] The endpoint that receives the notifications (email
  # address)
  # @return [Boolean] true if subscription was successfully created, false
  # otherwise
  def create_subscription(topic_arn, protocol, endpoint)
    @sns_client.subscribe(topic_arn: topic_arn, protocol: protocol, endpoint:
    endpoint)
    @logger.info("Subscription created successfully.")
    true
  rescue Aws::SNS::Errors::ServiceError => e
    @logger.error("Error while creating the subscription: #{e.message}")
    false
  end
end
```

```
end

# Main execution if the script is run directly
if $PROGRAM_NAME == __FILE__
  protocol = "email"
  endpoint = "EMAIL_ADDRESS" # Should be replaced with a real email address
  topic_arn = "TOPIC_ARN"    # Should be replaced with a real topic ARN

  sns_client = Aws::SNS::Client.new
  subscription_service = SubscriptionService.new(sns_client)

  @logger.info("Creating the subscription.")
  unless subscription_service.create_subscription(topic_arn, protocol, endpoint)
    @logger.error("Subscription creation failed. Stopping program.")
    exit 1
  end
end
end
```

- Para obter mais informações, consulte o [Guia do desenvolvedor do AWS SDK for Ruby](#).
- Para obter detalhes da API, consulte [Subscribe](#) na Referência da API AWS SDK for Ruby .

## Rust

### SDK para Rust

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

Inscreva um endereço de e-mail em um tópico.

```
async fn subscribe_and_publish(
  client: &Client,
  topic_arn: &str,
  email_address: &str,
) -> Result<(), Error> {
  println!("Receiving on topic with ARN: `{}`", topic_arn);
```

```
let rsp = client
    .subscribe()
    .topic_arn(topic_arn)
    .protocol("email")
    .endpoint(email_address)
    .send()
    .await?;

println!("Added a subscription: {:?}", rsp);

let rsp = client
    .publish()
    .topic_arn(topic_arn)
    .message("hello sns!")
    .send()
    .await?;

println!("Published message: {:?}", rsp);

Ok(())
}
```

- Para obter detalhes da API, consulte [Subscribe](#) na Referência da API AWS SDK para Rust.

## SAP ABAP

### SDK para SAP ABAP

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

Inscreva um endereço de e-mail em um tópico.

```
TRY.
    oo_result = lo_sns->subscribe(
returned for testing purposes."
        iv_topicarn = iv_topic_arn
        iv_protocol = 'email'
        "oo_result is
```

```
        iv_endpoint = iv_email_address
        iv_returnsubscriptionarn = abap_true
    ).
    MESSAGE 'Email address subscribed to SNS topic.' TYPE 'I'.
CATCH /aws1/cx_snsnotfoundexception.
    MESSAGE 'Topic does not exist.' TYPE 'E'.
CATCH /aws1/cx_snssubscriptionlmt00.
    MESSAGE 'Unable to create subscriptions. You have reached the maximum
number of subscriptions allowed.' TYPE 'E'.
ENDTRY.
```

- Para obter detalhes da API, consulte [Assinar](#) na Referência da API AWS SDK para SAP ABAP.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulte [Usando o Amazon SNS com um SDK AWS](#). Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

## Use **TagResource** com um AWS SDK ou CLI

Os exemplos de códigos a seguir mostram como usar `TagResource`.

### CLI

#### AWS CLI

Para adicionar uma tag a um tópico

O exemplo `tag-resource` a seguir adiciona uma tag de metadados ao tópico do Amazon SNS especificado.

```
aws sns tag-resource \
  --resource-arn arn:aws:sns:us-west-2:123456789012:MyTopic \
  --tags Key=Team,Value=Alpha
```

Este comando não produz saída.

- Para obter detalhes da API, consulte [TagResource](#) na Referência de AWS CLI Comandos.

## Java

## SDK para Java 2.x

 Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.sns.SnsClient;
import software.amazon.awssdk.services.sns.model.SnsException;
import software.amazon.awssdk.services.sns.model.Tag;
import software.amazon.awssdk.services.sns.model.TagResourceRequest;
import java.util.ArrayList;
import java.util.List;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
 * started.html
 */
public class AddTags {
    public static void main(String[] args) {
        final String usage = ""

                Usage:    <topicArn>

                Where:
                    topicArn - The ARN of the topic to which tags are added.

                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }
    }
}
```

```
String topicArn = args[0];
SnsClient snsClient = SnsClient.builder()
    .region(Region.US_EAST_1)
    .build();

addTopicTags(snsClient, topicArn);
snsClient.close();
}

public static void addTopicTags(SnsClient snsClient, String topicArn) {
    try {
        Tag tag = Tag.builder()
            .key("Team")
            .value("Development")
            .build();

        Tag tag2 = Tag.builder()
            .key("Environment")
            .value("Gamma")
            .build();

        List<Tag> tagList = new ArrayList<>();
        tagList.add(tag);
        tagList.add(tag2);

        TagResourceRequest tagResourceRequest = TagResourceRequest.builder()
            .resourceArn(topicArn)
            .tags(tagList)
            .build();

        snsClient.tagResource(tagResourceRequest);
        System.out.println("Tags have been added to " + topicArn);

    } catch (SnsException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}
}
```

- Para obter detalhes da API, consulte [TagResource](#) Referência AWS SDK for Java 2.x da API.

## Kotlin

### SDK para Kotlin

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
suspend fun addTopicTags(topicArn: String) {

    val tag = Tag {
        key = "Team"
        value = "Development"
    }

    val tag2 = Tag {
        key = "Environment"
        value = "Gamma"
    }

    val tagList = mutableListOf<Tag>()
    tagList.add(tag)
    tagList.add(tag2)

    val request = TagResourceRequest {
        resourceArn = topicArn
        tags = tagList
    }

    SnsClient { region = "us-east-1" }.use { snsClient ->
        snsClient.tagResource(request)
        println("Tags have been added to $topicArn")
    }
}
```

- Para obter detalhes da API, consulte a [TagResource](#) referência da API AWS SDK for Kotlin.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulte [Usando o Amazon SNS com um SDK AWS](#). Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

## Use **Unsubscribe** com um AWS SDK ou CLI

Os exemplos de códigos a seguir mostram como usar Unsubscribe.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação no contexto no seguinte exemplo de código:

- [Publicar mensagens em filas](#)

.NET

AWS SDK for .NET

### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

Cancele a assinatura de um tópico por meio de um ARN de assinatura.

```
/// <summary>
/// Unsubscribe from a topic by a subscription ARN.
/// </summary>
/// <param name="subscriptionArn">The ARN of the subscription.</param>
/// <returns>True if successful.</returns>
public async Task<bool> UnsubscribeByArn(string subscriptionArn)
{
    var unsubscribeResponse = await _amazonSNSClient.UnsubscribeAsync(
        new UnsubscribeRequest()
        {
            SubscriptionArn = subscriptionArn
        });
    return unsubscribeResponse.HttpStatusCode == HttpStatusCode.OK;
}
```

- Para obter detalhes da API, consulte [Unsubscribe](#) na Referência da API AWS SDK for .NET .

## C++

### SDK para C++

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
#!/ Delete a subscription to an Amazon Simple Notification Service (Amazon SNS)
topic.
/*!
 \param subscriptionARN: The Amazon Resource Name (ARN) for an Amazon SNS topic
 subscription.
 \param clientConfiguration: AWS client configuration.
 \return bool: Function succeeded.
 */
bool AwsDoc::SNS::unsubscribe(const Aws::String &subscriptionARN,
                             const Aws::Client::ClientConfiguration
&clientConfiguration) {
    Aws::SNS::SNSClient snsClient(clientConfiguration);

    Aws::SNS::Model::UnsubscribeRequest request;
    request.SetSubscriptionArn(subscriptionARN);

    const Aws::SNS::Model::UnsubscribeOutcome outcome =
snsClient.Unsubscribe(request);

    if (outcome.IsSuccess()) {
        std::cout << "Unsubscribed successfully " << std::endl;
    }
    else {
        std::cerr << "Error while unsubscribing " <<
outcome.GetError().GetMessage()
        << std::endl;
    }
}
```

```
    }  
  
    return outcome.IsSuccess();  
}
```

- Para obter detalhes da API, consulte [Unsubscribe](#) na Referência da API AWS SDK for C++ .

## CLI

### AWS CLI

Para cancelar a assinatura de um tópico

O exemplo `unsubscribe` a seguir exclui a assinatura especificada de um tópico.

```
aws sns unsubscribe \  
  --subscription-arn arn:aws:sns:us-west-2:0123456789012:my-  
  topic:8a21d249-4329-4871-acc6-7be709c6ea7f
```

Este comando não produz saída.

- Para obter detalhes da API, consulte [Unsubscribe](#) na Referência de comandos da AWS CLI .

## Java

### SDK para Java 2.x

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
import software.amazon.awssdk.regions.Region;  
import software.amazon.awssdk.services.sns.SnsClient;  
import software.amazon.awssdk.services.sns.model.SnsException;  
import software.amazon.awssdk.services.sns.model.UnsubscribeRequest;  
import software.amazon.awssdk.services.sns.model.UnsubscribeResponse;
```

```
/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */
public class Unsubscribe {
    public static void main(String[] args) {
        final String usage = ""

            Usage:    <subscriptionArn>

            Where:
                subscriptionArn - The ARN of the subscription to delete.
            "";

        if (args.length < 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String subscriptionArn = args[0];
        SnsClient snsClient = SnsClient.builder()
            .region(Region.US_EAST_1)
            .build();

        unSub(snsClient, subscriptionArn);
        snsClient.close();
    }

    public static void unSub(SnsClient snsClient, String subscriptionArn) {
        try {
            UnsubscribeRequest request = UnsubscribeRequest.builder()
                .subscriptionArn(subscriptionArn)
                .build();

            UnsubscribeResponse result = snsClient.unsubscribe(request);
            System.out.println("\n\nStatus was " +
                result.sdkHttpResponse().statusCode()
                + "\n\nSubscription was removed for " +
                request.subscriptionArn());
        }
    }
}
```

```
        } catch (SnsException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Para obter detalhes da API, consulte [Unsubscribe](#) na Referência da API AWS SDK for Java 2.x .

## JavaScript

### SDK para JavaScript (v3)

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

Crie o cliente em um módulo separado e exporte-o.

```
import { SNSClient } from "@aws-sdk/client-sns";

// The AWS Region can be provided here using the `region` property. If you leave
// it blank
// the SDK will default to the region set in your AWS config.
export const snsClient = new SNSClient({});
```

Importe o SDK e os módulos do cliente e chame a API.

```
import { UnsubscribeCommand } from "@aws-sdk/client-sns";
import { snsClient } from "../libs/snsClient.js";

/**
 * @param {string} subscriptionArn - The ARN of the subscription to cancel.
 */
const unsubscribe = async (
```

```
subscriptionArn = "arn:aws:sns:us-east-1:xxxxxxxxxxxx:mytopic:xxxxxxxx-xxxx-
xxxx-xxxx-xxxxxxxxxxxx",
) => {
  const response = await snsClient.send(
    new UnsubscribeCommand({
      SubscriptionArn: subscriptionArn,
    }),
  );
  console.log(response);
  // {
  //   '$metadata': {
  //     httpStatusCode: 200,
  //     requestId: '0178259a-9204-507c-b620-78a7570a44c6',
  //     extendedRequestId: undefined,
  //     cfId: undefined,
  //     attempts: 1,
  //     totalRetryDelay: 0
  //   }
  // }
  return response;
};
```

- Para obter mais informações, consulte o [Guia do desenvolvedor do AWS SDK for JavaScript](#).
- Para obter detalhes da API, consulte [Unsubscribe](#) na Referência da API AWS SDK for JavaScript .

## Kotlin

### SDK para Kotlin

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
suspend fun unSub(subscriptionArnVal: String) {

    val request = UnsubscribeRequest {
```

```
        subscriptionArn = subscriptionArnVal
    }

    SnsClient { region = "us-east-1" }.use { snsClient ->
        snsClient.unsubscribe(request)
        println("Subscription was removed for ${request.subscriptionArn}")
    }
}
```

- Para obter detalhes da API, consulte [Cancelar assinatura](#) na Referência da API AWS SDK para Kotlin.

## PHP

### SDK para PHP

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
require 'vendor/autoload.php';

use Aws\Exception\AwsException;
use Aws\Sns\SnsClient;

/**
 * Deletes a subscription to an Amazon SNS topic.
 *
 * This code expects that you have AWS credentials set up per:
 * https://docs.aws.amazon.com/sdk-for-php/v3/developer-guide/
 * guide_credentials.html
 */

$SnSClient = new SnsClient([
    'profile' => 'default',
    'region' => 'us-east-1',
    'version' => '2010-03-31'
```

```
]);

$subscription = 'arn:aws:sns:us-east-1:111122223333:MySubscription';

try {
    $result = $SnSClient->unsubscribe([
        'SubscriptionArn' => $subscription,
    ]);
    var_dump($result);
} catch (AwsException $e) {
    // output error message if fails
    error_log($e->getMessage());
}
```

- Para obter mais informações, consulte o [Guia do desenvolvedor do AWS SDK for PHP](#).
- Para obter detalhes da API, consulte [Unsubscribe](#) na Referência da API AWS SDK for PHP

## Python

### SDK para Python (Boto3)

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
class SnsWrapper:
    """Encapsulates Amazon SNS topic and subscription functions."""

    def __init__(self, sns_resource):
        """
        :param sns_resource: A Boto3 Amazon SNS resource.
        """
        self.sns_resource = sns_resource

    @staticmethod
```

```
def delete_subscription(subscription):
    """
    Unsubscribes and deletes a subscription.
    """
    try:
        subscription.delete()
        logger.info("Deleted subscription %s.", subscription.arn)
    except ClientError:
        logger.exception("Couldn't delete subscription %s.",
subscription.arn)
        raise
```

- Para obter detalhes da API, consulte [Unsubscribe](#) na Referência da API AWS SDK for Python (Boto3).

## SAP ABAP

### SDK para SAP ABAP

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
TRY.
  lo_sns->unsubscribe( iv_subscriptionarn = iv_subscription_arn ).
  MESSAGE 'Subscription deleted.' TYPE 'I'.
CATCH /aws1/cx_snsnotfoundexception.
  MESSAGE 'Subscription does not exist.' TYPE 'E'.
CATCH /aws1/cx_snsinvalidparameterex.
  MESSAGE 'Subscription with "PendingConfirmation" status cannot be
deleted/unsubscribed. Confirm subscription before performing unsubscribe
operation.' TYPE 'E'.
ENDTRY.
```

- Para obter detalhes da API, consulte [Cancelar assinatura](#) na Referência da API AWS SDK para SAP ABAP.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulte [Usando o Amazon SNS com um SDK AWS](#). Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

## Cenários para o Amazon SNS usando SDKs AWS

Os exemplos de código a seguir mostram como implementar cenários comuns no Amazon SNS com AWS SDKs. Esses cenários mostram como realizar tarefas específicas chamando várias funções no Amazon SNS. Cada cenário inclui um link para GitHub, onde você pode encontrar instruções sobre como configurar e executar o código.

### Exemplos

- [Crie um endpoint de plataforma para notificações push do Amazon SNS usando um SDK AWS](#)
- [Crie e publique em um tópico FIFO do Amazon SNS usando um SDK AWS](#)
- [Publique mensagens SMS em um tópico do Amazon SNS usando um SDK AWS](#)
- [Publique uma mensagem grande no Amazon SNS com o Amazon S3 usando um SDK AWS](#)
- [Publique uma mensagem de texto SMS do Amazon SNS usando um SDK AWS](#)
- [Publique mensagens do Amazon SNS nas filas do Amazon SQS usando um SDK AWS](#)

## Crie um endpoint de plataforma para notificações push do Amazon SNS usando um SDK AWS

Os exemplos de código a seguir mostram como criar um endpoint de plataforma para notificações por push do Amazon SNS.

### CLI

#### AWS CLI

Para criar um endpoint de aplicação de plataforma

O exemplo `create-platform-endpoint` a seguir cria um endpoint para a aplicação de plataforma especificada usando o token especificado.

```
aws sns create-platform-endpoint \  
  --platform-application-arn arn:aws:sns:us-west-2:123456789012:app/GCM/  
  MyApplication \  
  --platform-credentials-provider MyProvider
```

```
--token EXAMPLE12345...
```

**Saída:**

```
{
  "EndpointArn": "arn:aws:sns:us-west-2:1234567890:endpoint/GCM/
MyApplication/12345678-abcd-9012-efgh-345678901234"
}
```

**Java****SDK para Java 2.x**** Note**

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.sns.SnsClient;
import software.amazon.awssdk.services.sns.model.CreatePlatformEndpointRequest;
import software.amazon.awssdk.services.sns.model.CreatePlatformEndpointResponse;
import software.amazon.awssdk.services.sns.model.SnsException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 *
 * In addition, create a platform application using the AWS Management Console.
 * See this doc topic:
 *
 * https://docs.aws.amazon.com/sns/latest/dg/mobile-push-send-register.html
 *
 * Without the values created by following the previous link, this code examples
 * does not work.
```

```
*/

public class RegistrationExample {
    public static void main(String[] args) {
        final String usage = ""

            Usage:      <token> <platformApplicationArn>

            Where:
                token - The name of the FIFO topic.\s
                platformApplicationArn - The ARN value of platform
application. You can get this value from the AWS Management Console.\s
            """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String token = args[0];
        String platformApplicationArn = args[1];
        SnsClient snsClient = SnsClient.builder()
            .region(Region.US_EAST_1)
            .build();

        createEndpoint(snsClient, token, platformApplicationArn);
    }

    public static void createEndpoint(SnsClient snsClient, String token, String
platformApplicationArn) {
        System.out.println("Creating platform endpoint with token " + token);
        try {
            CreatePlatformEndpointRequest endpointRequest =
CreatePlatformEndpointRequest.builder()
                .token(token)
                .platformApplicationArn(platformApplicationArn)
                .build();

            CreatePlatformEndpointResponse response =
snsClient.createPlatformEndpoint(endpointRequest);
            System.out.println("The ARN of the endpoint is " +
response.endpointArn());
        } catch (SnsException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
        }
    }
}
```

```
        System.exit(1);
    }
}
}
```

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulte [Usando o Amazon SNS com um SDK AWS](#). Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

## Crie e publique em um tópico FIFO do Amazon SNS usando um SDK AWS

Os exemplos de código a seguir mostram como criar e publicar em um tópico FIFO do Amazon SNS.

### Java

#### SDK para Java 2.x

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

#### Esse exemplo

- cria um tópico FIFO do Amazon SNS, duas filas FIFO do Amazon SQS e uma fila padrão.
- inscreve as filas no tópico e publica a mensagem no tópico.

O [teste](#) verifica o recebimento da mensagem em cada fila. O [exemplo completo](#) também mostra a adição de políticas de acesso e exclui os recursos no final.

```
public class PriceUpdateExample {
    public final static SnsClient snsClient = SnsClient.create();
    public final static SqsClient sqsClient = SqsClient.create();

    public static void main(String[] args) {

        final String usage = "\n" +
            "Usage: " +
```

```
        "    <topicName> <wholesaleQueueFifoName> <retailQueueFifoName>
<analyticsQueueName>\n\n" +
        "Where:\n" +
        "    fifoTopicName - The name of the FIFO topic that you want to
create. \n\n" +
        "    wholesaleQueueARN - The name of a SQS FIFO queue that will be
created for the wholesale consumer. \n\n"
        +
        "    retailQueueARN - The name of a SQS FIFO queue that will
created for the retail consumer. \n\n" +
        "    analyticsQueueARN - The name of a SQS standard queue that
will be created for the analytics consumer. \n\n";
    if (args.length != 4) {
        System.out.println(usage);
        System.exit(1);
    }

    final String fifoTopicName = args[0];
    final String wholeSaleQueueName = args[1];
    final String retailQueueName = args[2];
    final String analyticsQueueName = args[3];

    // For convenience, the QueueData class holds metadata about a queue:
    // ARN, URL,
    // name and type.
    List<QueueData> queues = List.of(
        new QueueData(wholeSaleQueueName, QueueType.FIFO),
        new QueueData(retailQueueName, QueueType.FIFO),
        new QueueData(analyticsQueueName, QueueType.Standard));

    // Create queues.
    createQueues(queues);

    // Create a topic.
    String topicARN = createFIFOTopic(fifoTopicName);

    // Subscribe each queue to the topic.
    subscribeQueues(queues, topicARN);

    // Allow the newly created topic to send messages to the queues.
    addAccessPolicyToQueuesFINAL(queues, topicARN);

    // Publish a sample price update message with payload.
```

```
        publishPriceUpdate(topicARN, "{\"product\": 214, \"price\": 79.99}",
"Consumables");

        // Clean up resources.
        deleteSubscriptions(queues);
        deleteQueues(queues);
        deleteTopic(topicARN);
    }

    public static String createFIFOTopic(String topicName) {
        try {
            // Create a FIFO topic by using the SNS service client.
            Map<String, String> topicAttributes = Map.of(
                "FifoTopic", "true",
                "ContentBasedDeduplication", "false");

            CreateTopicRequest topicRequest = CreateTopicRequest.builder()
                .name(topicName)
                .attributes(topicAttributes)
                .build();

            CreateTopicResponse response = snsClient.createTopic(topicRequest);
            String topicArn = response.topicArn();
            System.out.println("The topic ARN is" + topicArn);

            return topicArn;

        } catch (SnsException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }

    public static void subscribeQueues(List<QueueData> queues, String topicARN) {
        queues.forEach(queue -> {
            SubscribeRequest subscribeRequest = SubscribeRequest.builder()
                .topicArn(topicARN)
                .endpoint(queue.queueARN)
                .protocol("sqs")
                .build();

            // Subscribe to the endpoint by using the SNS service client.
```

```
        // Only Amazon SQS queues can receive notifications from an Amazon
        SNS FIFO
        // topic.
        SubscribeResponse subscribeResponse =
snsClient.subscribe(subscribeRequest);
        System.out.println("The queue [" + queue.queueARN + "] subscribed to
the topic [" + topicARN + "]);
        queue.subscriptionARN = subscribeResponse.subscriptionArn();
    });
}

    public static void publishPriceUpdate(String topicArn, String payload, String
groupId) {

        try {
            // Create and publish a message that updates the wholesale price.
            String subject = "Price Update";
            String dedupId = UUID.randomUUID().toString();
            String attributeName = "business";
            String attributeValue = "wholesale";

            MessageAttributeValue msgAttValue = MessageAttributeValue.builder()
                .dataType("String")
                .stringValue(attributeValue)
                .build();

            Map<String, MessageAttributeValue> attributes = new HashMap<>();
            attributes.put(attributeName, msgAttValue);
            PublishRequest pubRequest = PublishRequest.builder()
                .topicArn(topicArn)
                .subject(subject)
                .message(payload)
                .messageGroupId(groupId)
                .messageDeduplicationId(dedupId)
                .messageAttributes(attributes)
                .build();

            final PublishResponse response = snsClient.publish(pubRequest);
            System.out.println(response.messageId());
            System.out.println(response.sequenceNumber());
            System.out.println("Message was published to " + topicArn);

        } catch (SnsException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
        }
    }
}
```

```
        System.exit(1);
    }
}
```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API AWS SDK for Java 2.x .
  - [CreateTopic](#)
  - [Publicar](#)
  - [Assinar](#)

## Python

### SDK para Python (Boto3)

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

Crie um tópico FIFO do Amazon SNS, inscreva filas padrão e FIFO do Amazon SQS no tópico e publique uma mensagem no tópico.

```
def usage_demo():
    """Shows how to subscribe queues to a FIFO topic."""
    print("-" * 88)
    print("Welcome to the `Subscribe queues to a FIFO topic` demo!")
    print("-" * 88)

    sns = boto3.resource("sns")
    sqs = boto3.resource("sqs")
    fifo_topic_wrapper = FifoTopicWrapper(sns)
    sns_wrapper = SnsWrapper(sns)

    prefix = "sqs-subscribe-demo-"
    queues = set()
    subscriptions = set()

    wholesale_queue = sqs.create_queue(
```

```
    QueueName=prefix + "wholesale.fifo",
    Attributes={
        "MaximumMessageSize": str(4096),
        "ReceiveMessageWaitTimeSeconds": str(10),
        "VisibilityTimeout": str(300),
        "FifoQueue": str(True),
        "ContentBasedDeduplication": str(True),
    },
)
queues.add(wholesale_queue)
print(f"Created FIFO queue with URL: {wholesale_queue.url}.")

retail_queue = sqs.create_queue(
    QueueName=prefix + "retail.fifo",
    Attributes={
        "MaximumMessageSize": str(4096),
        "ReceiveMessageWaitTimeSeconds": str(10),
        "VisibilityTimeout": str(300),
        "FifoQueue": str(True),
        "ContentBasedDeduplication": str(True),
    },
)
queues.add(retail_queue)
print(f"Created FIFO queue with URL: {retail_queue.url}.")

analytics_queue = sqs.create_queue(QueueName=prefix + "analytics",
Attributes={})
queues.add(analytics_queue)
print(f"Created standard queue with URL: {analytics_queue.url}.")

topic = fifo_topic_wrapper.create_fifo_topic("price-updates-topic.fifo")
print(f"Created FIFO topic: {topic.attributes['TopicArn']}.")

for q in queues:
    fifo_topic_wrapper.add_access_policy(q, topic.attributes["TopicArn"])

print(f"Added access policies for topic: {topic.attributes['TopicArn']}.")

for q in queues:
    sub = fifo_topic_wrapper.subscribe_queue_to_topic(
        topic, q.attributes["QueueArn"]
    )
    subscriptions.add(sub)
```

```
print(f"Subscribed queues to topic: {topic.attributes['TopicArn']}")

input("Press Enter to publish a message to the topic.")

message_id = fifo_topic_wrapper.publish_price_update(
    topic, '{"product": 214, "price": 79.99}', "Consumables"
)

print(f"Published price update with message ID: {message_id}.")

# Clean up the subscriptions, queues, and topic.
input("Press Enter to clean up resources.")
for s in subscriptions:
    sns_wrapper.delete_subscription(s)

sns_wrapper.delete_topic(topic)

for q in queues:
    fifo_topic_wrapper.delete_queue(q)

print(f"Deleted subscriptions, queues, and topic.")

print("Thanks for watching!")
print("-" * 88)

class FifoTopicWrapper:
    """Encapsulates Amazon SNS FIFO topic and subscription functions."""

    def __init__(self, sns_resource):
        """
        :param sns_resource: A Boto3 Amazon SNS resource.
        """
        self.sns_resource = sns_resource

    def create_fifo_topic(self, topic_name):
        """
        Create a FIFO topic.
        Topic names must be made up of only uppercase and lowercase ASCII
        letters,
        numbers, underscores, and hyphens, and must be between 1 and 256
        characters long.
        For a FIFO topic, the name must end with the .fifo suffix.
        """
```

```

:param topic_name: The name for the topic.
:return: The new topic.
"""
try:
    topic = self.sns_resource.create_topic(
        Name=topic_name,
        Attributes={
            "FifoTopic": str(True),
            "ContentBasedDeduplication": str(False),
        },
    )
    logger.info("Created FIFO topic with name=%s.", topic_name)
    return topic
except ClientError as error:
    logger.exception("Couldn't create topic with name=%s!", topic_name)
    raise error

@staticmethod
def add_access_policy(queue, topic_arn):
    """
    Add the necessary access policy to a queue, so
    it can receive messages from a topic.

    :param queue: The queue resource.
    :param topic_arn: The ARN of the topic.
    :return: None.
    """
    try:
        queue.set_attributes(
            Attributes={
                "Policy": json.dumps(
                    {
                        "Version": "2012-10-17",
                        "Statement": [
                            {
                                "Sid": "test-sid",
                                "Effect": "Allow",
                                "Principal": {"AWS": "*"},
                                "Action": "SQS:SendMessage",
                                "Resource": queue.attributes["QueueArn"],
                                "Condition": {
                                    "ArnLike": {"aws:SourceArn": topic_arn}
                                }
                            }
                        ]
                    }
                )
            }
        )
    
```

```
        },
    ],
}

)

)
    logger.info("Added trust policy to the queue.")
except ClientError as error:
    logger.exception("Couldn't add trust policy to the queue!")
    raise error

@staticmethod
def subscribe_queue_to_topic(topic, queue_arn):
    """
    Subscribe a queue to a topic.

    :param topic: The topic resource.
    :param queue_arn: The ARN of the queue.
    :return: The subscription resource.
    """
    try:
        subscription = topic.subscribe(
            Protocol="sqs",
            Endpoint=queue_arn,
        )
        logger.info("The queue is subscribed to the topic.")
        return subscription
    except ClientError as error:
        logger.exception("Couldn't subscribe queue to topic!")
        raise error

@staticmethod
def publish_price_update(topic, payload, group_id):
    """
    Compose and publish a message that updates the wholesale price.

    :param topic: The topic to publish to.
    :param payload: The message to publish.
    :param group_id: The group ID for the message.
    :return: The ID of the message.
    """
```

```

    try:
        att_dict = {"business": {"DataType": "String", "StringValue":
"wholesale"}}
        dedup_id = uuid.uuid4()
        response = topic.publish(
            Subject="Price Update",
            Message=payload,
            MessageAttributes=att_dict,
            MessageGroupId=group_id,
            MessageDeduplicationId=str(dedup_id),
        )
        message_id = response["MessageId"]
        logger.info("Published message to topic %s.", topic.arn)
    except ClientError as error:
        logger.exception("Couldn't publish message to topic %s.", topic.arn)
        raise error
    return message_id

@staticmethod
def delete_queue(queue):
    """
    Removes an SQS queue. When run against an AWS account, it can take up to
    60 seconds before the queue is actually deleted.

    :param queue: The queue to delete.
    :return: None
    """
    try:
        queue.delete()
        logger.info("Deleted queue with URL=%s.", queue.url)
    except ClientError as error:
        logger.exception("Couldn't delete queue with URL=%s!", queue.url)
        raise error

```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API AWS SDK para Python (Boto3).
  - [CreateTopic](#)

- [Publicar](#)
- [Assinar](#)

## SAP ABAP

### SDK para SAP ABAP

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

Crie um tópico FIFO, inscreva uma fila FIFO do Amazon SQS no tópico e publique uma mensagem em um tópico do Amazon SNS.

```

" Creates a FIFO topic. "
DATA lt_tpc_attributes TYPE /aws1/
cl_snstopicattrsmmap_w=>tt_topicattributesmap.
DATA ls_tpc_attributes TYPE /aws1/
cl_snstopicattrsmmap_w=>ts_topicattributesmap_maprow.
ls_tpc_attributes-key = 'FifoTopic'.
ls_tpc_attributes-value = NEW /aws1/cl_snstopicattrsmmap_w( iv_value =
'true' ).
INSERT ls_tpc_attributes INTO TABLE lt_tpc_attributes.

TRY.
  DATA(lo_create_result) = lo_sns->createtopic(
    iv_name = iv_topic_name
    it_attributes = lt_tpc_attributes
  ).
  DATA(lv_topic_arn) = lo_create_result->get_topicarn( ).
  ov_topic_arn = lv_topic_arn.
  "
ov_topic_arn is returned for testing purposes. "
  MESSAGE 'FIFO topic created' TYPE 'I'.
CATCH /aws1/cx_snstopiclimitexcdex.
  MESSAGE 'Unable to create more topics. You have reached the maximum
number of topics allowed.' TYPE 'E'.
ENDTRY.

```

```

" Subscribes an endpoint to an Amazon Simple Notification Service (Amazon
SNS) topic. "
" Only Amazon Simple Queue Service (Amazon SQS) FIFO queues can be subscribed
to an SNS FIFO topic. "
TRY.
    DATA(lo_subscribe_result) = lo_sns->subscribe(
        iv_topicarn = lv_topic_arn
        iv_protocol = 'sqs'
        iv_endpoint = iv_queue_arn
    ).
    DATA(lv_subscription_arn) = lo_subscribe_result->get_subscriptionarn( ).
    ov_subscription_arn = lv_subscription_arn.
ov_subscription_arn is returned for testing purposes. "
    MESSAGE 'SQS queue was subscribed to SNS topic.' TYPE 'I'.
CATCH /aws1/cx_snsnotfoundexception.
    MESSAGE 'Topic does not exist.' TYPE 'E'.
CATCH /aws1/cx_snssubscriptionlmt00.
    MESSAGE 'Unable to create subscriptions. You have reached the maximum
number of subscriptions allowed.' TYPE 'E'.
ENDTRY.

" Publish message to SNS topic. "
TRY.
    DATA lt_msg_attributes TYPE /aws1/
cl_snsmessageattrvalue=>tt_messageattributemap.
    DATA ls_msg_attributes TYPE /aws1/
cl_snsmessageattrvalue=>ts_messageattributemap_maprow.
    ls_msg_attributes-key = 'Importance'.
    ls_msg_attributes-value = NEW /aws1/cl_snsmessageattrvalue( iv_datatype =
'String' iv_stringvalue = 'High' ).
    INSERT ls_msg_attributes INTO TABLE lt_msg_attributes.

    DATA(lo_result) = lo_sns->publish(
        iv_topicarn = lv_topic_arn
        iv_message = 'The price of your mobile plan has been increased from
$19 to $23'
        iv_subject = 'Changes to mobile plan'
        iv_messagegroupid = 'Update-2'
        iv_messagededuplicationid = 'Update-2.1'
        it_messageattributes = lt_msg_attributes
    ).
    ov_message_id = lo_result->get_messageid( ).
ov_message_id is returned for testing purposes. "
    MESSAGE 'Message was published to SNS topic.' TYPE 'I'.

```

```
CATCH /aws1/cx_snsnotfoundexception.  
    MESSAGE 'Topic does not exist.' TYPE 'E'.  
ENDTRY.
```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API AWS SDK para SAP ABAP.
  - [CreateTopic](#)
  - [Publicar](#)
  - [Assinar](#)

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulte [Usando o Amazon SNS com um SDK AWS](#). Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

## Publique mensagens SMS em um tópico do Amazon SNS usando um SDK AWS

O exemplo de código a seguir mostra como:

- Criar um tópico do Amazon SNS.
- Inscrever números de telefone no tópico.
- Publicar mensagens SMS no tópico para que todos os números de telefone inscritos recebam a mensagem de uma só vez.

### Java

#### SDK para Java 2.x

##### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

Criar um tópico e retorne seu ARN.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.sns.SnsClient;
import software.amazon.awssdk.services.sns.model.CreateTopicRequest;
import software.amazon.awssdk.services.sns.model.CreateTopicResponse;
import software.amazon.awssdk.services.sns.model.SnsException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */
public class CreateTopic {
    public static void main(String[] args) {
        final String usage = ""

            Usage:    <topicName>

            Where:
                topicName - The name of the topic to create (for example,
mytopic).

            """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String topicName = args[0];
        System.out.println("Creating a topic with name: " + topicName);
        SnsClient snsClient = SnsClient.builder()
            .region(Region.US_EAST_1)
            .build();

        String arnVal = createSNSTopic(snsClient, topicName);
        System.out.println("The topic ARN is" + arnVal);
        snsClient.close();
    }
}
```

```
public static String createSNSTopic(SnsClient snsClient, String topicName) {
    CreateTopicResponse result;
    try {
        CreateTopicRequest request = CreateTopicRequest.builder()
            .name(topicName)
            .build();

        result = snsClient.createTopic(request);
        return result.topicArn();

    } catch (SnsException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
    return "";
}
```

Inscreva um endpoint em um tópico.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.sns.SnsClient;
import software.amazon.awssdk.services.sns.model.SnsException;
import software.amazon.awssdk.services.sns.model.SubscribeRequest;
import software.amazon.awssdk.services.sns.model.SubscribeResponse;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */
public class SubscribeTextSMS {
    public static void main(String[] args) {
        final String usage = ""

            Usage:    <topicArn> <phoneNumber>

            Where:
```

```
        topicArn - The ARN of the topic to subscribe.
        phoneNumber - A mobile phone number that receives
notifications (for example, +1XXX5550100).
        """;

    if (args.length < 2) {
        System.out.println(usage);
        System.exit(1);
    }

    String topicArn = args[0];
    String phoneNumber = args[1];
    SnsClient snsClient = SnsClient.builder()
        .region(Region.US_EAST_1)
        .build();

    subTextSNS(snsClient, topicArn, phoneNumber);
    snsClient.close();
}

public static void subTextSNS(SnsClient snsClient, String topicArn, String
phoneNumber) {
    try {
        SubscribeRequest request = SubscribeRequest.builder()
            .protocol("sms")
            .endpoint(phoneNumber)
            .returnSubscriptionArn(true)
            .topicArn(topicArn)
            .build();

        SubscribeResponse result = snsClient.subscribe(request);
        System.out.println("Subscription ARN: " + result.subscriptionArn() +
"\n\n Status is "
            + result.sdkHttpResponse().statusCode());

    } catch (SnsException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}
}
```

Defina atributos na mensagem, como o ID do remetente, o preço máximo e seu tipo. Os atributos de mensagem são opcionais.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.sns.SnsClient;
import software.amazon.awssdk.services.sns.model.SetSmsAttributesRequest;
import software.amazon.awssdk.services.sns.model.SetSmsAttributesResponse;
import software.amazon.awssdk.services.sns.model.SnsException;
import java.util.HashMap;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */
public class SetSMSAttributes {
    public static void main(String[] args) {
        HashMap<String, String> attributes = new HashMap<>(1);
        attributes.put("DefaultSMSType", "Transactional");
        attributes.put("UsageReportS3Bucket", "janbucket");

        SnsClient snsClient = SnsClient.builder()
            .region(Region.US_EAST_1)
            .build();
        setSMSAttributes(snsClient, attributes);
        snsClient.close();
    }

    public static void setSMSAttributes(SnsClient snsClient, HashMap<String,
String> attributes) {
        try {
            SetSmsAttributesRequest request = SetSmsAttributesRequest.builder()
                .attributes(attributes)
                .build();

            SetSmsAttributesResponse result =
snsClient.setSMSAttributes(request);
            System.out.println("Set default Attributes to " + attributes + ".
Status was ")

```

```
        + result.sdkHttpResponse().statusCode());

    } catch (SnsException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}
}
```

Publique uma mensagem em um tópico. A mensagem é enviada para todos os assinantes.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.sns.SnsClient;
import software.amazon.awssdk.services.sns.model.PublishRequest;
import software.amazon.awssdk.services.sns.model.PublishResponse;
import software.amazon.awssdk.services.sns.model.SnsException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */
public class PublishTextSMS {
    public static void main(String[] args) {
        final String usage = ""

                Usage:    <message> <phoneNumber>

                Where:
                    message - The message text to send.
                    phoneNumber - The mobile phone number to which a message is
sent (for example, +1XXX5550100).\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }
    }
}
```

```
String message = args[0];
String phoneNumber = args[1];
SnsClient snsClient = SnsClient.builder()
    .region(Region.US_EAST_1)
    .build();
pubTextSMS(snsClient, message, phoneNumber);
snsClient.close();
}

public static void pubTextSMS(SnsClient snsClient, String message, String
phoneNumber) {
    try {
        PublishRequest request = PublishRequest.builder()
            .message(message)
            .phoneNumber(phoneNumber)
            .build();

        PublishResponse result = snsClient.publish(request);
        System.out
            .println(result.messageId() + " Message sent. Status was " +
result.sdkHttpResponse().statusCode());

    } catch (SnsException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}
}
```

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulte [Usando o Amazon SNS com um SDK AWS](#). Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

## Publique uma mensagem grande no Amazon SNS com o Amazon S3 usando um SDK AWS

O exemplo de código a seguir mostra como publicar uma mensagem grande no Amazon SNS usando o Amazon S3 para armazenar a carga útil da mensagem.

## Java

### SDK para Java 1.x

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

Para publicar uma mensagem grande, use a Amazon SNS Extended Client Library for Java. A mensagem que você envia faz referência a um objeto do Amazon S3 que contém o conteúdo real da mensagem.

```
import com.amazon.sqs.javamessaging.AmazonSQSExtendedClient;
import com.amazon.sqs.javamessaging.ExtendedClientConfiguration;
import com.amazonaws.regions.Region;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.sns.AmazonSNS;
import com.amazonaws.services.sns.AmazonSNSClientBuilder;
import com.amazonaws.services.sns.model.CreateTopicRequest;
import com.amazonaws.services.sns.model.PublishRequest;
import com.amazonaws.services.sns.model.SetSubscriptionAttributesRequest;
import com.amazonaws.services.sns.util.Topics;
import com.amazonaws.services.sqs.AmazonSQS;
import com.amazonaws.services.sqs.AmazonSQSClientBuilder;
import com.amazonaws.services.sqs.model.CreateQueueRequest;
import com.amazonaws.services.sqs.model.ReceiveMessageResult;
import software.amazon.sns.AmazonSNSExtendedClient;
import software.amazon.sns.SNSExtendedClientConfiguration;

public class Example {

    public static void main(String[] args) {
        final String BUCKET_NAME = "extended-client-bucket";
        final String TOPIC_NAME = "extended-client-topic";
        final String QUEUE_NAME = "extended-client-queue";
        final Regions region = Regions.DEFAULT_REGION;
```

```
// Message threshold controls the maximum message size that will
be allowed to
// be published
// through SNS using the extended client. Payload of messages
exceeding this
// value will be stored in
// S3. The default value of this parameter is 256 KB which is the
maximum
// message size in SNS (and SQS).
final int EXTENDED_STORAGE_MESSAGE_SIZE_THRESHOLD = 32;

// Initialize SNS, SQS and S3 clients
final AmazonSNS snsClient =
AmazonSNSClientBuilder.standard().withRegion(region).build();
final AmazonSQS sqsClient =
AmazonSQSClientBuilder.standard().withRegion(region).build();
final AmazonS3 s3Client =
AmazonS3ClientBuilder.standard().withRegion(region).build();

// Create bucket, topic, queue and subscription
s3Client.createBucket(BUCKET_NAME);
final String topicArn = snsClient.createTopic(
    new
CreateTopicRequest().withName(TOPIC_NAME)).getTopicArn();
final String queueUrl = sqsClient.createQueue(
    new
CreateQueueRequest().withQueueName(QueueName)).getQueueUrl();
final String subscriptionArn = Topics.subscribeQueue(
    snsClient, sqsClient, topicArn, queueUrl);

// To read message content stored in S3 transparently through SQS
extended
// client,
// set the RawMessageDelivery subscription attribute to TRUE
final SetSubscriptionAttributesRequest
subscriptionAttributesRequest = new SetSubscriptionAttributesRequest();

subscriptionAttributesRequest.setSubscriptionArn(subscriptionArn);

subscriptionAttributesRequest.setAttributeName("RawMessageDelivery");
subscriptionAttributesRequest.setAttributeValue("TRUE");

snsClient.setSubscriptionAttributes(subscriptionAttributesRequest);
```

```
        // Initialize SNS extended client
        // PayloadSizeThreshold triggers message content storage in S3
when the
        // threshold is exceeded
        // To store all messages content in S3, use AlwaysThroughS3 flag
        final SNSExtendedClientConfiguration
snsExtendedClientConfiguration = new SNSExtendedClientConfiguration()
                                .withPayloadSupportEnabled(s3Client, BUCKET_NAME)

        .withPayloadSizeThreshold(EXTENDED_STORAGE_MESSAGE_SIZE_THRESHOLD);
        final AmazonSNSExtendedClient snsExtendedClient = new
AmazonSNSExtendedClient(snsClient,
                        snsExtendedClientConfiguration);

        // Publish message via SNS with storage in S3
        final String message = "This message is stored in S3 as it
exceeds the threshold of 32 bytes set above.";
        snsExtendedClient.publish(topicArn, message);

        // Initialize SQS extended client
        final ExtendedClientConfiguration sqsExtendedClientConfiguration
= new ExtendedClientConfiguration()
                                .withPayloadSupportEnabled(s3Client,
BUCKET_NAME);
        final AmazonSQSExtendedClient sqsExtendedClient = new
AmazonSQSExtendedClient(sqsClient,
                        sqsExtendedClientConfiguration);

        // Read the message from the queue
        final ReceiveMessageResult result =
sqsExtendedClient.receiveMessage(queueUrl);
        System.out.println("Received message is " +
result.getMessages().get(0).getBody());
    }
}
```

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulte [Usando o Amazon SNS com um SDK AWS](#). Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

# Publique uma mensagem de texto SMS do Amazon SNS usando um SDK AWS

Os exemplos de código a seguir mostram como publicar mensagens SMS usando o Amazon SNS.

.NET

AWS SDK for .NET

## Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
namespace SNSMessageExample
{
    using System;
    using System.Threading.Tasks;
    using Amazon;
    using Amazon.SimpleNotificationService;
    using Amazon.SimpleNotificationService.Model;

    public class SNSMessage
    {
        private AmazonSimpleNotificationServiceClient snsClient;

        /// <summary>
        /// Initializes a new instance of the <see cref="SNSMessage"/> class.
        /// Constructs a new SNSMessage object initializing the Amazon Simple
        /// Notification Service (Amazon SNS) client using the supplied
        /// Region endpoint.
        /// </summary>
        /// <param name="regionEndpoint">The Amazon Region endpoint to use in
        /// sending test messages with this object.</param>
        public SNSMessage(RegionEndpoint regionEndpoint)
        {
            snsClient = new
AmazonSimpleNotificationServiceClient(regionEndpoint);
        }

        /// <summary>
```

```
    /// Sends the SMS message passed in the text parameter to the phone
    number
    /// in phoneNum.
    /// </summary>
    /// <param name="phoneNum">The ten-digit phone number to which the text
    /// message will be sent.</param>
    /// <param name="text">The text of the message to send.</param>
    /// <returns>Async task.</returns>
    public async Task SendTextMessageAsync(string phoneNum, string text)
    {
        if (string.IsNullOrEmpty(phoneNum) || string.IsNullOrEmpty(text))
        {
            return;
        }

        // Now actually send the message.
        var request = new PublishRequest
        {
            Message = text,
            PhoneNumber = phoneNum,
        };

        try
        {
            var response = await snsClient.PublishAsync(request);
        }
        catch (Exception ex)
        {
            Console.WriteLine($"Error sending message: {ex}");
        }
    }
}
```

- Para obter detalhes da API, consulte [Publish](#) na Referência da API AWS SDK for .NET .

## C++

## SDK para C++

 Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
/**
 * Publish SMS: use Amazon Simple Notification Service (Amazon SNS) to send an
 * SMS text message to a phone number.
 * Note: This requires additional AWS configuration prior to running example.
 *
 * NOTE: When you start using Amazon SNS to send SMS messages, your AWS account
 * is in the SMS sandbox and you can only
 * use verified destination phone numbers. See https://docs.aws.amazon.com/sns/
 * latest/dg/sns-sms-sandbox.html.
 * NOTE: If destination is in the US, you also have an additional restriction
 * that you have use a dedicated
 * origination ID (phone number). You can request an origination number using
 * Amazon Pinpoint for a fee.
 * See https://aws.amazon.com/blogs/compute/provisioning-and-using-10dlc-
 * origination-numbers-with-amazon-sns/
 * for more information.
 *
 * <phone_number_value> input parameter uses E.164 format.
 * For example, in United States, this input value should be of the form:
 * +12223334444
 */

//! Send an SMS text message to a phone number.
/*!
 \param message: The message to publish.
 \param phoneNumber: The phone number of the recipient in E.164 format.
 \param clientConfiguration: AWS client configuration.
 \return bool: Function succeeded.
 */
bool AwsDoc::SNS::publishSms(const Aws::String &message,
                             const Aws::String &phoneNumber,
```

```
const Aws::Client::ClientConfiguration
&clientConfiguration) {
    Aws::SNS::SNSClient snsClient(clientConfiguration);

    Aws::SNS::Model::PublishRequest request;
    request.SetMessage(message);
    request.SetPhoneNumber(phoneNumber);

    const Aws::SNS::Model::PublishOutcome outcome = snsClient.Publish(request);

    if (outcome.IsSuccess()) {
        std::cout << "Message published successfully with message id, '"
            << outcome.GetResult().GetMessageId() << "'."
            << std::endl;
    }
    else {
        std::cerr << "Error while publishing message "
            << outcome.GetError().GetMessage()
            << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Para obter detalhes da API, consulte [Publish](#) na Referência da API AWS SDK for C++ .

## Java

### SDK para Java 2.x

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.sns.SnsClient;
import software.amazon.awssdk.services.sns.model.PublishRequest;
import software.amazon.awssdk.services.sns.model.PublishResponse;
import software.amazon.awssdk.services.sns.model.SnsException;
```

```
/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */
public class PublishTextSMS {
    public static void main(String[] args) {
        final String usage = ""

            Usage:    <message> <phoneNumber>

            Where:
                message - The message text to send.
                phoneNumber - The mobile phone number to which a message is
sent (for example, +1XXX5550100).\s
            """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String message = args[0];
        String phoneNumber = args[1];
        SnsClient snsClient = SnsClient.builder()
            .region(Region.US_EAST_1)
            .build();
        pubTextSMS(snsClient, message, phoneNumber);
        snsClient.close();
    }

    public static void pubTextSMS(SnsClient snsClient, String message, String
phoneNumber) {
        try {
            PublishRequest request = PublishRequest.builder()
                .message(message)
                .phoneNumber(phoneNumber)
                .build();
```

```
        PublishResponse result = snsClient.publish(request);
        System.out
            .println(result.messageId() + " Message sent. Status was " +
result.sdkHttpResponse().statusCode());

    } catch (SnsException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}
}
```

- Para obter detalhes da API, consulte [Publish](#) na Referência da API AWS SDK for Java 2.x .

## Kotlin

### SDK para Kotlin

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
suspend fun pubTextSMS(messageVal: String?, phoneNumberVal: String?) {

    val request = PublishRequest {
        message = messageVal
        phoneNumber = phoneNumberVal
    }

    SnsClient { region = "us-east-1" }.use { snsClient ->
        val result = snsClient.publish(request)
        println("${result.messageId} message sent.")
    }
}
```

- Para obter detalhes da API, consulte [Publish](#) na Referência da API AWS SDK para Kotlin.

## PHP

## SDK para PHP

 Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
require 'vendor/autoload.php';

use Aws\Exception\AwsException;
use Aws\Sns\SnsClient;

/**
 * Sends a text message (SMS message) directly to a phone number using Amazon
 * SNS.
 *
 * This code expects that you have AWS credentials set up per:
 * https://docs.aws.amazon.com/sdk-for-php/v3/developer-guide/
 * guide_credentials.html
 */

$SnSClient = new SnsClient([
    'profile' => 'default',
    'region' => 'us-east-1',
    'version' => '2010-03-31'
]);

$message = 'This message is sent from a Amazon SNS code sample.';
$phone = '+1XXX5550100';

try {
    $result = $SnSClient->publish([
        'Message' => $message,
        'PhoneNumber' => $phone,
    ]);
    var_dump($result);
} catch (AwsException $e) {
    // output error message if fails
}
```

```
error_log($e->getMessage());
}
```

- Para obter mais informações, consulte o [Guia do desenvolvedor do AWS SDK for PHP](#).
- Para obter detalhes da API, consulte [Publish](#) na Referência da API AWS SDK for PHP .

## Python

### SDK para Python (Boto3)

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
class SnsWrapper:
    """Encapsulates Amazon SNS topic and subscription functions."""

    def __init__(self, sns_resource):
        """
        :param sns_resource: A Boto3 Amazon SNS resource.
        """
        self.sns_resource = sns_resource

    def publish_text_message(self, phone_number, message):
        """
        Publishes a text message directly to a phone number without need for a
        subscription.

        :param phone_number: The phone number that receives the message. This
        must be
                               in E.164 format. For example, a United States phone
                               number might be +12065550101.
        :param message: The message to send.
        :return: The ID of the message.
        """
        try:
```

```
        response = self.sns_resource.meta.client.publish(
            PhoneNumber=phone_number, Message=message
        )
        message_id = response["MessageId"]
        logger.info("Published message to %s.", phone_number)
    except ClientError:
        logger.exception("Couldn't publish message to %s.", phone_number)
        raise
    else:
        return message_id
```

- Para obter detalhes da API, consulte [Publish](#) na Referência da API AWS SDK for Python (Boto3).

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulte [Usando o Amazon SNS com um SDK AWS](#). Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

## Publique mensagens do Amazon SNS nas filas do Amazon SQS usando um SDK AWS

Os exemplos de código a seguir mostram como:

- Criar um tópico (FIFO ou não FIFO).
- Assinar várias filas no tópico com a opção de aplicar um filtro.
- Publicar mensagens no tópico.
- Pesquisar as filas para ver as mensagens recebidas.

### .NET

#### AWS SDK for .NET

##### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

## Execute um cenário interativo em um prompt de comando.

```
/// <summary>
/// Console application to run a workflow scenario for topics and queues.
/// </summary>
public static class TopicsAndQueues
{
    private static bool _useFifoTopic = false;
    private static bool _useContentBasedDeduplication = false;
    private static string _topicName = null!;
    private static string _topicArn = null!;

    private static readonly int _queueCount = 2;
    private static readonly string[] _queueUrls = new string[_queueCount];
    private static readonly string[] _subscriptionArns = new string[_queueCount];
    private static readonly string[] _tones = { "cheerful", "funny", "serious",
"sincere" };
    public static SNSWrapper SnsWrapper { get; set; } = null!;
    public static SQSWrapper SqsWrapper { get; set; } = null!;
    public static bool UseConsole { get; set; } = true;
    static async Task Main(string[] args)
    {
        // Set up dependency injection for Amazon EventBridge.
        using var host = Host.CreateDefaultBuilder(args)
            .ConfigureLogging(logging =>
                logging.AddFilter("System", LogLevel.Debug)
                    .AddFilter<DebugLoggerProvider>("Microsoft",
LogLevel.Information)
                    .AddFilter<ConsoleLoggerProvider>("Microsoft",
LogLevel.Trace))
            .ConfigureServices((_, services) =>
                services.AddAWSService<IAmazonSQS>()
                    .AddAWSService<IAmazonSimpleNotificationService>()
                    .AddTransient<SNSWrapper>()
                    .AddTransient<SQSWrapper>()
            )
            .Build();

        ServicesSetup(host);
        PrintDescription();

        await RunScenario();
    }
}
```

```
/// <summary>
/// Populate the services for use within the console application.
/// </summary>
/// <param name="host">The services host.</param>
private static void ServicesSetup(IHost host)
{
    SnsWrapper = host.Services.GetRequiredService<SNSWrapper>();
    SqsWrapper = host.Services.GetRequiredService<SQSWrapper>();
}

/// <summary>
/// Run the scenario for working with topics and queues.
/// </summary>
/// <returns>True if successful.</returns>
public static async Task<bool> RunScenario()
{
    try
    {
        await SetupTopic();

        await SetupQueues();

        await PublishMessages();

        foreach (var queueUrl in _queueUrls)
        {
            var messages = await PollForMessages(queueUrl);
            if (messages.Any())
            {
                await DeleteMessages(queueUrl, messages);
            }
        }
        await CleanupResources();

        Console.WriteLine("Messaging with topics and queues workflow is
complete.");
        return true;
    }
    catch (Exception ex)
    {
        Console.WriteLine(new string('-', 80));
        Console.WriteLine($"There was a problem running the scenario:
{ex.Message}");
    }
}
```

```
        await CleanupResources();
        Console.WriteLine(new string('-', 80));
        return false;
    }
}

/// <summary>
/// Print a description for the tasks in the workflow.
/// </summary>
/// <returns>Async task.</returns>
private static void PrintDescription()
{
    Console.WriteLine(new string('-', 80));
    Console.WriteLine($"Welcome to messaging with topics and queues.");

    Console.WriteLine(new string('-', 80));
    Console.WriteLine($"In this workflow, you will create an SNS topic and
subscribe {_queueCount} SQS queues to the topic." +
        $"{\r\nYou can select from several options for
configuring the topic and the subscriptions for the 2 queues." +
        $"{\r\nYou can then post to the topic and see the
results in the queues.\r\n}");

    Console.WriteLine(new string('-', 80));
}

/// <summary>
/// Set up the SNS topic to be used with the queues.
/// </summary>
/// <returns>Async task.</returns>
private static async Task<string> SetupTopic()
{
    Console.WriteLine(new string('-', 80));
    Console.WriteLine($"SNS topics can be configured as FIFO (First-In-First-
Out)." +
        $"{\r\nFIFO topics deliver messages in order and support
deduplication and message filtering." +
        $"{\r\nYou can then post to the topic and see the
results in the queues.\r\n}");

    _useFifoTopic = GetYesNoResponse("Would you like to work with FIFO
topics?");

    if (_useFifoTopic)
```

```
    {
        Console.WriteLine(new string('-', 80));
        _topicName = GetUserResponse("Enter a name for your SNS topic: ",
"example-topic");
        Console.WriteLine(
            "Because you have selected a FIFO topic, '.fifo' must be appended
to the topic name.\r\n");

        Console.WriteLine(new string('-', 80));
        Console.WriteLine($"Because you have chosen a FIFO topic,
deduplication is supported." +
            "\r\nDeduplication IDs are either set in the
message or automatically generated " +
            "\r\nfrom content using a hash function.\r\n" +
            "\r\nIf a message is successfully published to an
SNS FIFO topic, any message " +
            "\r\npublished and determined to have the same
deduplication ID, " +
            "\r\nwithin the five-minute deduplication
interval, is accepted but not delivered.\r\n" +
            "\r\nFor more information about deduplication, " +
            "\r\nsee https://docs.aws.amazon.com/sns/latest/
dg/fifo-message-dedup.html.");

        _useContentBasedDeduplication = GetYesNoResponse("Use content-based
deduplication instead of entering a deduplication ID?");
        Console.WriteLine(new string('-', 80));
    }

    _topicArn = await SnsWrapper.CreateTopicWithName(_topicName,
_useFifoTopic, _useContentBasedDeduplication);

    Console.WriteLine($"Your new topic with the name {_topicName}" +
        "\r\nand Amazon Resource Name (ARN) {_topicArn}" +
        "\r\nhas been created.\r\n");

    Console.WriteLine(new string('-', 80));
    return _topicArn;
}

/// <summary>
/// Set up the queues.
/// </summary>
/// <returns>Async task.</returns>
```

```
private static async Task SetupQueues()
{
    Console.WriteLine(new string('-', 80));
    Console.WriteLine($"Now you will create {_queueCount} Amazon Simple Queue
Service (Amazon SQS) queues to subscribe to the topic.");

    // Repeat this section for each queue.
    for (int i = 0; i < _queueCount; i++)
    {
        var queueName = GetUserResponse("Enter a name for an Amazon SQS
queue: ", $"example-queue-{i}");
        if (_useFifoTopic)
        {
            // Only explain this once.
            if (i == 0)
            {
                Console.WriteLine(
                    "Because you have selected a FIFO topic, '.fifo' must be
appended to the queue name.");
            }

            var queueUrl = await SqsWrapper.CreateQueueWithName(queueName,
_useFifoTopic);

            _queueUrls[i] = queueUrl;

            Console.WriteLine($"Your new queue with the name {queueName}" +
                $"{"\r\n"}and queue URL {queueUrl}" +
                $"{"\r\n"}has been created.{"\r\n"}");

            if (i == 0)
            {
                Console.WriteLine(
                    $"The queue URL is used to retrieve the queue ARN,{"\r\n"} +
                    $"which is used to create a subscription.");
                Console.WriteLine(new string('-', 80));
            }

            var queueArn = await SqsWrapper.GetQueueArnByUrl(queueUrl);

            if (i == 0)
            {
                Console.WriteLine(
```

```
        $"An AWS Identity and Access Management (IAM) policy must
be attached to an SQS queue, enabling it to receive\r\n" +
        $"messages from an SNS topic");
    }

    await SqsWrapper.SetQueuePolicyForTopic(queueArn, _topicArn,
queueUrl);

    await SetupFilters(i, queueArn, queueName);
}
}

Console.WriteLine(new string('-', 80));
}

/// <summary>
/// Set up filters with user options for a queue.
/// </summary>
/// <param name="queueCount">The number of this queue.</param>
/// <param name="queueArn">The ARN of the queue.</param>
/// <param name="queueName">The name of the queue.</param>
/// <returns>Async Task.</returns>
public static async Task SetupFilters(int queueCount, string queueArn, string
queueName)
{
    if (_useFifoTopic)
    {
        Console.WriteLine(new string('-', 80));
        // Only explain this once.
        if (queueCount == 0)
        {
            Console.WriteLine(
                "Subscriptions to a FIFO topic can have filters." +
                "If you add a filter to this subscription, then only the
filtered messages " +
                "will be received in the queue.");

            Console.WriteLine(
                "For information about message filtering, " +
                "see https://docs.aws.amazon.com/sns/latest/dg/sns-message-
filtering.html");

            Console.WriteLine(
                "For this example, you can filter messages by a " +
```

```
        "TONE attribute.");
    }

    var useFilter = GetYesNoResponse($"Filter messages for {queueName}'s
subscription to the topic?");

    string? filterPolicy = null;
    if (useFilter)
    {
        filterPolicy = CreateFilterPolicy();
    }
    var subscriptionArn = await
SnsWrapper.SubscribeTopicWithFilter(_topicArn, filterPolicy,
        queueArn);
    _subscriptionArns[queueCount] = subscriptionArn;

    Console.WriteLine(
        $"The queue {queueName} has been subscribed to the topic
{_topicName} " +
        $"with the subscription ARN {subscriptionArn}");
    Console.WriteLine(new string('-', 80));
    }
}

/// <summary>
/// Use user input to create a filter policy for a subscription.
/// </summary>
/// <returns>The serialized filter policy.</returns>
public static string CreateFilterPolicy()
{
    Console.WriteLine(new string('-', 80));
    Console.WriteLine(
        $"You can filter messages by one or more of the following" +
        $"TONE attributes.");

    List<string> filterSelections = new List<string>();

    var selectionNumber = 0;
    do
    {
        Console.WriteLine(
            $"Enter a number to add a TONE filter, or enter 0 to stop adding
filters.");
        for (int i = 0; i < _tones.Length; i++)
```

```
        {
            Console.WriteLine($"\\t{i + 1}. {_tones[i]}");
        }

        var selection = GetUserResponse("", filterSelections.Any() ? "0" :
"1");
        int.TryParse(selection, out selectionNumber);
        if (selectionNumber > 0 && !
filterSelections.Contains(_tones[selectionNumber - 1]))
        {
            filterSelections.Add(_tones[selectionNumber - 1]);
        }
    } while (selectionNumber != 0);

    var filters = new Dictionary<string, List<string>>
    {
        { "tone", filterSelections }
    };
    string filterPolicy = JsonSerializer.Serialize(filters);
    return filterPolicy;
}

/// <summary>
/// Publish messages using user settings.
/// </summary>
/// <returns>Async task.</returns>
public static async Task PublishMessages()
{
    Console.WriteLine("Now we can publish messages.");

    var keepSendingMessages = true;
    string? deduplicationId = null;
    string? toneAttribute = null;
    while (keepSendingMessages)
    {
        Console.WriteLine();
        var message = GetUserResponse("Enter a message to publish.", "This is
a sample message");

        if (_useFifoTopic)
        {
            Console.WriteLine("Because you are using a FIFO topic, you must
set a message group ID." +
```

```
        "\r\nAll messages within the same group will be
received in the order " +
        "they were published.");

        Console.WriteLine();
        var messageId = GetUserResponse("Enter a message group ID
for this message:", "1");

        if (!_useContentBasedDeduplication)
        {
            Console.WriteLine("Because you are not using content-based
deduplication, " +
                "you must enter a deduplication ID.");

            Console.WriteLine("Enter a deduplication ID for this
message.");
            deduplicationId = GetUserResponse("Enter a deduplication ID
for this message.", "1");
        }

        if (GetYesNoResponse("Add an attribute to this message?"))
        {
            Console.WriteLine("Enter a number for an attribute.");
            for (int i = 0; i < _tones.Length; i++)
            {
                Console.WriteLine($"{i + 1}. {_tones[i]}");
            }

            var selection = GetUserResponse("", "1");
            int.TryParse(selection, out var selectionNumber);

            if (selectionNumber > 0 && selectionNumber < _tones.Length)
            {
                toneAttribute = _tones[selectionNumber - 1];
            }
        }

        var messageId = await SnsWrapper.PublishToTopicWithAttribute(
            _topicArn, message, "tone", toneAttribute, deduplicationId,
messageGroupId);

        Console.WriteLine($"Message published with id {messageID}.");
    }
}
```

```
        keepSendingMessages = GetYesNoResponse("Send another message?",
false);
    }
}

/// <summary>
/// Poll for the published messages to see the results of the user's choices.
/// </summary>
/// <returns>Async task.</returns>
public static async Task<List<Message>> PollForMessages(string queueUrl)
{
    Console.WriteLine(new string('-', 80));
    Console.WriteLine($"Now the SQS queue at {queueUrl} will be polled to
retrieve the messages." +
        "\r\nPress any key to continue.");
    if (UseConsole)
    {
        Console.ReadLine();
    }

    var moreMessages = true;
    var messages = new List<Message>();
    while (moreMessages)
    {
        var newMessages = await SqsWrapper.ReceiveMessagesByUrl(queueUrl,
10);

        moreMessages = newMessages.Any();
        if (moreMessages)
        {
            messages.AddRange(newMessages);
        }
    }

    Console.WriteLine($"{messages.Count} message(s) were received by the
queue at {queueUrl}.");

    foreach (var message in messages)
    {
        Console.WriteLine("\tMessage:" +
            $"{"\n\t{message.Body}");
    }

    Console.WriteLine(new string('-', 80));
}
```

```
        return messages;
    }

    /// <summary>
    /// Delete the message using handles in a batch.
    /// </summary>
    /// <returns>Async task.</returns>
    public static async Task DeleteMessages(string queueUrl, List<Message>
messages)
    {
        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Now we can delete the messages in this queue in a
batch.");
        await SqsWrapper.DeleteMessageBatchByUrl(queueUrl, messages);
        Console.WriteLine(new string('-', 80));
    }

    /// <summary>
    /// Clean up the resources from the scenario.
    /// </summary>
    /// <returns>Async task.</returns>
    private static async Task CleanupResources()
    {
        Console.WriteLine(new string('-', 80));
        Console.WriteLine($"Clean up resources.");

        try
        {
            foreach (var queueUrl in _queueUrls)
            {
                if (!string.IsNullOrEmpty(queueUrl))
                {
                    var deleteQueue =
                        GetYesNoResponse($"Delete queue with url {queueUrl}?");
                    if (deleteQueue)
                    {
                        await SqsWrapper.DeleteQueueByUrl(queueUrl);
                    }
                }
            }

            foreach (var subscriptionArn in _subscriptionArns)
            {
                if (!string.IsNullOrEmpty(subscriptionArn))
```

```
        {
            await SnsWrapper.UnsubscribeByArn(subscriptionArn);
        }
    }

    var deleteTopic = GetYesNoResponse($"Delete topic {_topicName}?");
    if (deleteTopic)
    {
        await SnsWrapper.DeleteTopicByArn(_topicArn);
    }
}
catch (Exception ex)
{
    Console.WriteLine($"Unable to clean up resources. Here's why:
{ex.Message}.");
}

Console.WriteLine(new string('-', 80));
}

/// <summary>
/// Helper method to get a yes or no response from the user.
/// </summary>
/// <param name="question">The question string to print on the console.</
param>
/// <param name="defaultAnswer">Optional default answer to use.</param>
/// <returns>True if the user responds with a yes.</returns>
private static bool GetYesNoResponse(string question, bool defaultAnswer =
true)
{
    if (UseConsole)
    {
        Console.WriteLine(question);
        var ynResponse = Console.ReadLine();
        var response = ynResponse != null &&
            ynResponse.Equals("y",
                StringComparison.InvariantCultureIgnoreCase);
        return response;
    }
    // If not using the console, use the default.
    return defaultAnswer;
}

/// <summary>
```

```
    /// Helper method to get a string response from the user through the console.
    /// </summary>
    /// <param name="question">The question string to print on the console.</
param>
    /// <param name="defaultAnswer">Optional default answer to use.</param>
    /// <returns>True if the user responds with a yes.</returns>
    private static string GetUserResponse(string question, string defaultAnswer)
    {
        if (UseConsole)
        {
            var response = "";
            while (string.IsNullOrEmpty(response))
            {
                Console.WriteLine(question);
                response = Console.ReadLine();
            }
            return response;
        }
        // If not using the console, use the default.
        return defaultAnswer;
    }
}
```

Crie uma classe que envolva operações do Amazon SQS.

```
/// <summary>
/// Wrapper for Amazon Simple Queue Service (SQS) operations.
/// </summary>
public class SQSWrapper
{
    private readonly IAmazonSQS _amazonSQSClient;

    /// <summary>
    /// Constructor for the Amazon SQS wrapper.
    /// </summary>
    /// <param name="amazonSQS">The injected Amazon SQS client.</param>
    public SQSWrapper(IAmazonSQS amazonSQS)
    {
        _amazonSQSClient = amazonSQS;
    }
}
```

```
/// <summary>
/// Create a queue with a specific name.
/// </summary>
/// <param name="queueName">The name for the queue.</param>
/// <param name="useFifoQueue">True to use a FIFO queue.</param>
/// <returns>The url for the queue.</returns>
public async Task<string> CreateQueueWithName(string queueName, bool
useFifoQueue)
{
    int maxMessage = 256 * 1024;
    var queueAttributes = new Dictionary<string, string>
    {
        {
            QueueAttributeName.MaximumMessageSize,
            maxMessage.ToString()
        }
    };

    var createQueueRequest = new CreateQueueRequest()
    {
        QueueName = queueName,
        Attributes = queueAttributes
    };

    if (useFifoQueue)
    {
        // Update the name if it is not correct for a FIFO queue.
        if (!queueName.EndsWith(".fifo"))
        {
            createQueueRequest.QueueName = queueName + ".fifo";
        }

        // Add an attribute for a FIFO queue.
        createQueueRequest.Attributes.Add(
            QueueAttributeName.FifoQueue, "true");
    }

    var createResponse = await _amazonSQSClient.CreateQueueAsync(
        new CreateQueueRequest()
        {
            QueueName = queueName
        });
    return createResponse.QueueUrl;
}
```

```
/// <summary>
/// Get the ARN for a queue from its URL.
/// </summary>
/// <param name="queueUrl">The URL of the queue.</param>
/// <returns>The ARN of the queue.</returns>
public async Task<string> GetQueueArnByUrl(string queueUrl)
{
    var getAttributesRequest = new GetQueueAttributesRequest()
    {
        QueueUrl = queueUrl,
        AttributeNames = new List<string>() { QueueAttributeName.QueueArn }
    };

    var getAttributesResponse = await
        _amazonSQSClient.GetQueueAttributesAsync(
            getAttributesRequest);

    return getAttributesResponse.QueueARN;
}

/// <summary>
/// Set the policy attribute of a queue for a topic.
/// </summary>
/// <param name="queueArn">The ARN of the queue.</param>
/// <param name="topicArn">The ARN of the topic.</param>
/// <param name="queueUrl">The url for the queue.</param>
/// <returns>True if successful.</returns>
public async Task<bool> SetQueuePolicyForTopic(string queueArn, string
topicArn, string queueUrl)
{
    var queuePolicy = "{" +
        "\"Version\": \"2012-10-17\"," +
        "\"Statement\": [{" +
            "\"Effect\": \"Allow\"," +
            "\"Principal\": {" +
                "\"Service\": " +
                    "\"sns.amazonaws.com\"" +
                "}," +
            "\"Action\": \"sqs:SendMessage\"," +
            "\"Resource\": \"{queueArn}\"," +
            "\"Condition\": {" +
                "\"ArnEquals\": {" +
```

```

                                                                    $"\"aws:SourceArn\":
\"{topicArn}\" +
                                                                    "]" +
                                                                    "]" +
                                                                    "]" +
                                                                    "];
    var attributesResponse = await _amazonSQSClient.SetQueueAttributesAsync(
        new SetQueueAttributesRequest()
        {
            QueueUrl = queueUrl,
            Attributes = new Dictionary<string, string>() { { "Policy",
queuePolicy } }
        });
    return attributesResponse.HttpStatusCode == HttpStatusCode.OK;
}

/// <summary>
/// Receive messages from a queue by its URL.
/// </summary>
/// <param name="queueUrl">The url of the queue.</param>
/// <returns>The list of messages.</returns>
public async Task<List<Message>> ReceiveMessagesByUrl(string queueUrl, int
maxMessages)
{
    // Setting WaitTimeSeconds to non-zero enables long polling.
    // For information about long polling, see
    // https://docs.aws.amazon.com/AWSSimpleQueueService/latest/
SQSDeveloperGuide/sqs-short-and-long-polling.html
    var messageResponse = await _amazonSQSClient.ReceiveMessageAsync(
        new ReceiveMessageRequest()
        {
            QueueUrl = queueUrl,
            MaxNumberOfMessages = maxMessages,
            WaitTimeSeconds = 1
        });
    return messageResponse.Messages;
}

/// <summary>
/// Delete a batch of messages from a queue by its url.
/// </summary>
/// <param name="queueUrl">The url of the queue.</param>
/// <returns>True if successful.</returns>

```

```
public async Task<bool> DeleteMessageBatchByUrl(string queueUrl,
List<Message> messages)
{
    var deleteRequest = new DeleteMessageBatchRequest()
    {
        QueueUrl = queueUrl,
        Entries = new List<DeleteMessageBatchRequestEntry>()
    };
    foreach (var message in messages)
    {
        deleteRequest.Entries.Add(new DeleteMessageBatchRequestEntry()
        {
            ReceiptHandle = message.ReceiptHandle,
            Id = message.MessageId
        });
    }

    var deleteResponse = await
_amazonSQSClient.DeleteMessageBatchAsync(deleteRequest);

    return deleteResponse.Failed.Any();
}

/// <summary>
/// Delete a queue by its URL.
/// </summary>
/// <param name="queueUrl">The url of the queue.</param>
/// <returns>True if successful.</returns>
public async Task<bool> DeleteQueueByUrl(string queueUrl)
{
    var deleteResponse = await _amazonSQSClient.DeleteQueueAsync(
        new DeleteQueueRequest()
        {
            QueueUrl = queueUrl
        });
    return deleteResponse.HttpStatusCode == HttpStatusCode.OK;
}
}
```

Crie uma classe que envolva operações do Amazon SNS.

```
/// <summary>
/// Wrapper for Amazon Simple Notification Service (SNS) operations.
/// </summary>
public class SNSWrapper
{
    private readonly IAmazonSimpleNotificationService _amazonSNSClient;

    /// <summary>
    /// Constructor for the Amazon SNS wrapper.
    /// </summary>
    /// <param name="amazonSNS">The injected Amazon SNS client.</param>
    public SNSWrapper(IAmazonSimpleNotificationService amazonSNS)
    {
        _amazonSNSClient = amazonSNS;
    }

    /// <summary>
    /// Create a new topic with a name and specific FIFO and de-duplication
    attributes.
    /// </summary>
    /// <param name="topicName">The name for the topic.</param>
    /// <param name="useFifoTopic">True to use a FIFO topic.</param>
    /// <param name="useContentBasedDeduplication">True to use content-based de-
    duplication.</param>
    /// <returns>The ARN of the new topic.</returns>
    public async Task<string> CreateTopicWithName(string topicName, bool
    useFifoTopic, bool useContentBasedDeduplication)
    {
        var createTopicRequest = new CreateTopicRequest()
        {
            Name = topicName,
        };

        if (useFifoTopic)
        {
            // Update the name if it is not correct for a FIFO topic.
            if (!topicName.EndsWith(".fifo"))
            {
                createTopicRequest.Name = topicName + ".fifo";
            }

            // Add the attributes from the method parameters.
            createTopicRequest.Attributes = new Dictionary<string, string>
            {
```

```
        { "FifoTopic", "true" }
    };
    if (useContentBasedDeduplication)
    {
        createTopicRequest.Attributes.Add("ContentBasedDeduplication",
"true");
    }
}

var createResponse = await
_amazonSNSClient.CreateTopicAsync(createTopicRequest);
return createResponse.TopicArn;
}

/// <summary>
/// Subscribe a queue to a topic with optional filters.
/// </summary>
/// <param name="topicArn">The ARN of the topic.</param>
/// <param name="useFifoTopic">The optional filtering policy for the
subscription.</param>
/// <param name="queueArn">The ARN of the queue.</param>
/// <returns>The ARN of the new subscription.</returns>
public async Task<string> SubscribeTopicWithFilter(string topicArn, string?
filterPolicy, string queueArn)
{
    var subscribeRequest = new SubscribeRequest()
    {
        TopicArn = topicArn,
        Protocol = "sqs",
        Endpoint = queueArn
    };

    if (!string.IsNullOrEmpty(filterPolicy))
    {
        subscribeRequest.Attributes = new Dictionary<string, string>
{ { "FilterPolicy", filterPolicy } };
    }

    var subscribeResponse = await
_amazonSNSClient.SubscribeAsync(subscribeRequest);
    return subscribeResponse.SubscriptionArn;
}

/// <summary>
```

```
    /// Publish a message to a topic with an attribute and optional deduplication
    and group IDs.
    /// </summary>
    /// <param name="topicArn">The ARN of the topic.</param>
    /// <param name="message">The message to publish.</param>
    /// <param name="attributeName">The optional attribute for the message.</
param>
    /// <param name="attributeValue">The optional attribute value for the
    message.</param>
    /// <param name="deduplicationId">The optional deduplication ID for the
    message.</param>
    /// <param name="groupId">The optional group ID for the message.</param>
    /// <returns>The ID of the message published.</returns>
    public async Task<string> PublishToTopicWithAttribute(
        string topicArn,
        string message,
        string? attributeName = null,
        string? attributeValue = null,
        string? deduplicationId = null,
        string? groupId = null)
    {
        var publishRequest = new PublishRequest()
        {
            TopicArn = topicArn,
            Message = message,
            MessageDeduplicationId = deduplicationId,
            MessageGroupId = groupId
        };

        if (attributeValue != null)
        {
            // Add the string attribute if it exists.
            publishRequest.MessageAttributes =
                new Dictionary<string, MessageAttributeValue>
                {
                    { attributeName!, new MessageAttributeValue() { StringValue =
attributeValue, DataType = "String" } }
                };
        }

        var publishResponse = await
        _amazonSNSClient.PublishAsync(publishRequest);
        return publishResponse.MessageId;
    }
}
```

```
/// <summary>
/// Unsubscribe from a topic by a subscription ARN.
/// </summary>
/// <param name="subscriptionArn">The ARN of the subscription.</param>
/// <returns>True if successful.</returns>
public async Task<bool> UnsubscribeByArn(string subscriptionArn)
{
    var unsubscribeResponse = await _amazonSNSClient.UnsubscribeAsync(
        new UnsubscribeRequest()
        {
            SubscriptionArn = subscriptionArn
        });
    return unsubscribeResponse.HttpStatusCode == HttpStatusCode.OK;
}

/// <summary>
/// Delete a topic by its topic ARN.
/// </summary>
/// <param name="topicArn">The ARN of the topic.</param>
/// <returns>True if successful.</returns>
public async Task<bool> DeleteTopicByArn(string topicArn)
{
    var deleteResponse = await _amazonSNSClient.DeleteTopicAsync(
        new DeleteTopicRequest()
        {
            TopicArn = topicArn
        });
    return deleteResponse.HttpStatusCode == HttpStatusCode.OK;
}
}
```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API AWS SDK for .NET .
  - [CreateQueue](#)
  - [CreateTopic](#)
  - [DeleteMessageBatch](#)
  - [DeleteQueue](#)
  - [DeleteTopic](#)

- [GetQueueAttributes](#)
- [Publicar](#)
- [ReceiveMessage](#)
- [SetQueueAttributes](#)
- [Assinar](#)
- [Cancelar assinatura](#)

## C++

### SDK para C++

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
Aws::Client::ClientConfiguration clientConfig;
// Optional: Set to the AWS Region (overrides config file).
// clientConfig.region = "us-east-1";

//! Workflow for messaging with topics and queues using Amazon SNS and Amazon
SQS.
/*!
 \param clientConfig Aws client configuration.
 \return bool: Successful completion.
 */
bool AwsDoc::TopicsAndQueues::messagingWithTopicsAndQueues(
    const Aws::Client::ClientConfiguration &clientConfiguration) {
    std::cout << "Welcome to messaging with topics and queues." << std::endl;
    printAsterisksLine();
    std::cout << "In this workflow, you will create an SNS topic and subscribe "
        << NUMBER_OF_QUEUES <<
        " SQS queues to the topic." << std::endl;
    std::cout
        << "You can select from several options for configuring the topic and
the subscriptions for the "
        << NUMBER_OF_QUEUES << " queues." << std::endl;
```

```
std::cout << "You can then post to the topic and see the results in the
queues."
    << std::endl;

Aws::SNS::SNSClient snsClient(clientConfiguration);

printAsterisksLine();

std::cout << "SNS topics can be configured as FIFO (First-In-First-Out)."
    << std::endl;
std::cout
    << "FIFO topics deliver messages in order and support deduplication
and message filtering."
    << std::endl;
bool isFifoTopic = askYesNoQuestion(
    "Would you like to work with FIFO topics? (y/n) ");

bool contentBasedDeduplication = false;
Aws::String topicName;
if (isFifoTopic) {
    printAsterisksLine();
    std::cout << "Because you have chosen a FIFO topic, deduplication is
supported."
        << std::endl;
    std::cout
        << "Deduplication IDs are either set in the message or
automatically generated "
        << "from content using a hash function." << std::endl;
    std::cout
        << "If a message is successfully published to an SNS FIFO topic,
any message "
        << "published and determined to have the same deduplication ID, "
        << std::endl;
    std::cout
        << "within the five-minute deduplication interval, is accepted
but not delivered."
        << std::endl;
    std::cout
        << "For more information about deduplication, "
        << "see https://docs.aws.amazon.com/sns/latest/dg/fifo-message-
dedup.html."
        << std::endl;
    contentBasedDeduplication = askYesNoQuestion(
```

```
        "Use content-based deduplication instead of entering a
deduplication ID? (y/n) ");
    }

    printAsterisksLine();

    Aws::SQS::SQSClient sqsClient(clientConfiguration);
    Aws::Vector<Aws::String> queueURLS;
    Aws::Vector<Aws::String> subscriptionARNs;

    Aws::String topicARN;
    {
        topicName = askQuestion("Enter a name for your SNS topic. ");

        // 1. Create an Amazon SNS topic, either FIFO or non-FIFO.
        Aws::SNS::Model::CreateTopicRequest request;

        if (isFifoTopic) {
            request.AddAttributes("FifoTopic", "true");
            if (contentBasedDeduplication) {
                request.AddAttributes("ContentBasedDeduplication", "true");
            }
            topicName = topicName + FIFO_SUFFIX;

            std::cout
                << "Because you have selected a FIFO topic, '.fifo' must be
appended to the topic name."
                << std::endl;
        }

        request.SetName(topicName);

        Aws::SNS::Model::CreateTopicOutcome outcome =
snsClient.CreateTopic(request);

        if (outcome.IsSuccess()) {
            topicARN = outcome.GetResult().GetTopicArn();
            std::cout << "Your new topic with the name '" << topicName
                << "' and the topic Amazon Resource Name (ARN) " <<
std::endl;
            std::cout << "'" << topicARN << "' has been created." << std::endl;
        }
        else {
```

```

        std::cerr << "Error with TopicsAndQueues::CreateTopic. "
                << outcome.GetError().GetMessage()
                << std::endl;

        cleanUp(topicARN,
                queueURLS,
                subscriptionARNS,
                snsClient,
                sqsClient);

        return false;
    }
}

printAsterisksLine();

std::cout << "Now you will create " << NUMBER_OF_QUEUES
          << " SQS queues to subscribe to the topic." << std::endl;
Aws::Vector<Aws::String> queueNames;
bool filteringMessages = false;
bool first = true;
for (int i = 1; i <= NUMBER_OF_QUEUES; ++i) {
    Aws::String queueURL;
    Aws::String queueName;
    {
        printAsterisksLine();
        std::ostringstream ostream;
        ostream << "Enter a name for " << (first ? "an" : "the next")
                << " SQS queue. ";
        queueName = askQuestion(ostream.str());

        // 2. Create an SQS queue.
        Aws::SQS::Model::CreateQueueRequest request;
        if (isFifoTopic) {
            request.AddAttributes(Aws::SQS::Model::QueueAttributeName::FifoQueue,
                                "true");
            queueName = queueName + FIFO_SUFFIX;

            if (first) // Only explain this once.
            {
                std::cout
                    << "Because you are creating a FIFO SQS queue,
'.fifo' must "

```

```
        << "be appended to the queue name." << std::endl;
    }
}

request.SetQueueName(queueName);
queueNames.push_back(queueName);

Aws::SQS::Model::CreateQueueOutcome outcome =
    sqsClient.CreateQueue(request);

if (outcome.IsSuccess()) {
    queueURL = outcome.GetResult().GetQueueUrl();
    std::cout << "Your new SQS queue with the name '" << queueName
        << "' and the queue URL " << std::endl;
    std::cout << "'" << queueURL << "' has been created." <<
std::endl;
}
else {
    std::cerr << "Error with SQS::CreateQueue. "
        << outcome.GetError().GetMessage()
        << std::endl;

    cleanUp(topicARN,
        queueURLS,
        subscriptionARNS,
        snsClient,
        sqsClient);

    return false;
}
}
queueURLS.push_back(queueURL);

if (first) // Only explain this once.
{
    std::cout
        << "The queue URL is used to retrieve the queue ARN, which is
"
        << "used to create a subscription." << std::endl;
}

Aws::String queueARN;
{
    // 3. Get the SQS queue ARN attribute.
```

```
    Aws::SQS::Model::GetQueueAttributesRequest request;
    request.SetQueueUrl(queueURL);

request.AddAttributeNames(Aws::SQS::Model::QueueAttributeName::QueueArn);

    Aws::SQS::Model::GetQueueAttributesOutcome outcome =
        sqsClient.GetQueueAttributes(request);

    if (outcome.IsSuccess()) {
        const Aws::Map<Aws::SQS::Model::QueueAttributeName, Aws::String>
&attributes =
            outcome.GetResult().GetAttributes();
        const auto &iter = attributes.find(
            Aws::SQS::Model::QueueAttributeName::QueueArn);
        if (iter != attributes.end()) {
            queueARN = iter->second;
            std::cout << "The queue ARN '" << queueARN
                << "' has been retrieved."
                << std::endl;
        }
        else {
            std::cerr
                << "Error ARN attribute not returned by
GetQueueAttribute."
                << std::endl;

            cleanUp(topicARN,
                queueURLS,
                subscriptionARNS,
                snsClient,
                sqsClient);

            return false;
        }
    }
    else {
        std::cerr << "Error with SQS::GetQueueAttributes. "
            << outcome.GetError().GetMessage()
            << std::endl;

        cleanUp(topicARN,
            queueURLS,
            subscriptionARNS,
            snsClient,
```

```
        sqsClient);

        return false;
    }
}

if (first) {
    std::cout
        << "An IAM policy must be attached to an SQS queue, enabling
it to receive "
        << "messages from an SNS topic." << std::endl;
}

{
    // 4. Set the SQS queue policy attribute with a policy enabling the
receipt of SNS messages.
    Aws::SQS::Model::SetQueueAttributesRequest request;
    request.SetQueueUrl(queueURL);
    Aws::String policy = createPolicyForQueue(queueARN, topicARN);
    request.AddAttributes(Aws::SQS::Model::QueueAttributeName::Policy,
        policy);

    Aws::SQS::Model::SetQueueAttributesOutcome outcome =
        sqsClient.SetQueueAttributes(request);

    if (outcome.IsSuccess()) {
        std::cout << "The attributes for the queue '" << queueName
            << "' were successfully updated." << std::endl;
    }
    else {
        std::cerr << "Error with SQS::SetQueueAttributes. "
            << outcome.GetError().GetMessage()
            << std::endl;

        cleanUp(topicARN,
            queueURLS,
            subscriptionARNS,
            snsClient,
            sqsClient);

        return false;
    }
}
```

```

printAsterisksLine();

{
    // 5. Subscribe the SQS queue to the SNS topic.
    Aws::SNS::Model::SubscribeRequest request;
    request.SetTopicArn(topicARN);
    request.SetProtocol("sqs");
    request.SetEndpoint(queueARN);
    if (isFifoTopic) {
        if (first) {
            std::cout << "Subscriptions to a FIFO topic can have
filters."
                << std::endl;
            std::cout
                << "If you add a filter to this subscription, then
only the filtered messages "
                << "will be received in the queue." << std::endl;
            std::cout << "For information about message filtering, "
                << "see https://docs.aws.amazon.com/sns/latest/dg/
sns-message-filtering.html"
                << std::endl;
            std::cout << "For this example, you can filter messages by a
\"
                << TONE_ATTRIBUTE << "\" attribute." << std::endl;
        }

        std::ostringstream ostream;
        ostream << "Filter messages for \"" << queueName
            << "\"'s subscription to the topic \""
            << topicName << "\"? (y/n)";

        // Add filter if user answers yes.
        if (askYesNoQuestion(ostream.str())) {
            Aws::String jsonPolicy = getFilterPolicyFromUser();
            if (!jsonPolicy.empty()) {
                filteringMessages = true;

                std::cout << "This is the filter policy for this
subscription."
                    << std::endl;
                std::cout << jsonPolicy << std::endl;

                request.AddAttributes("FilterPolicy", jsonPolicy);
            }

```

```

        else {
            std::cout
                << "Because you did not select any attributes, no
filter "
                << "will be added to this subscription." <<
std::endl;
        }
    } // if (isFifoTopic)
    Aws::SNS::Model::SubscribeOutcome outcome =
snsClient.Subscribe(request);

    if (outcome.IsSuccess()) {
        Aws::String subscriptionARN =
outcome.GetResult().GetSubscriptionArn();
        std::cout << "The queue '" << queueName
            << "' has been subscribed to the topic '"
            << "'" << topicName << "'" << std::endl;
        std::cout << "with the subscription ARN '" << subscriptionARN <<
"."
            << std::endl;
        subscriptionARNS.push_back(subscriptionARN);
    }
    else {
        std::cerr << "Error with TopicsAndQueues::Subscribe. "
            << outcome.GetError().GetMessage()
            << std::endl;

        cleanUp(topicARN,
            queueURLS,
            subscriptionARNS,
            snsClient,
            sqsClient);

        return false;
    }
}

first = false;
}

first = true;
do {
    printAsterisksLine();

```

```

// 6. Publish a message to the SNS topic.
Aws::SNS::Model::PublishRequest request;
request.SetTopicArn(topicARN);
Aws::String message = askQuestion("Enter a message text to publish. ");
request.SetMessage(message);
if (isFifoTopic) {
    if (first) {
        std::cout
            << "Because you are using a FIFO topic, you must set a
message group ID."
            << std::endl;
        std::cout
            << "All messages within the same group will be received
in the "
            << "order they were published." << std::endl;
    }
    Aws::String messageGroupID = askQuestion(
        "Enter a message group ID for this message. ");
    request.SetMessageGroupId(messageGroupID);
    if (!contentBasedDeduplication) {
        if (first) {
            std::cout
                << "Because you are not using content-based
deduplication, "
                << "you must enter a deduplication ID." << std::endl;
        }
        Aws::String deduplicationID = askQuestion(
            "Enter a deduplication ID for this message. ");
        request.SetMessageDeduplicationId(deduplicationID);
    }
}

if (filteringMessages && askYesNoQuestion(
    "Add an attribute to this message? (y/n) ")) {
    for (size_t i = 0; i < TONES.size(); ++i) {
        std::cout << " " << (i + 1) << ". " << TONES[i] << std::endl;
    }
    int selection = askQuestionForIntRange(
        "Enter a number for an attribute. ",
        1, static_cast<int>(TONES.size()));
    Aws::SNS::Model::MessageAttributeValue messageAttributeValue;
    messageAttributeValue.SetDataType("String");
    messageAttributeValue.SetStringValue(TONES[selection - 1]);
}

```

```
        request.AddMessageAttributes(TONE_ATTRIBUTE, messageAttributeValue);
    }

    Aws::SNS::Model::PublishOutcome outcome = snsClient.Publish(request);

    if (outcome.IsSuccess()) {
        std::cout << "Your message was successfully published." << std::endl;
    }
    else {
        std::cerr << "Error with TopicsAndQueues::Publish. "
                  << outcome.GetError().GetMessage()
                  << std::endl;

        cleanUp(topicARN,
                queueURLS,
                subscriptionARNS,
                snsClient,
                sqsClient);

        return false;
    }

    first = false;
} while (askYesNoQuestion("Post another message? (y/n) "));

printAsterisksLine();

std::cout << "Now the SQS queue will be polled to retrieve the messages."
          << std::endl;
askQuestion("Press any key to continue...", alwaysTrueTest);

for (size_t i = 0; i < queueURLS.size(); ++i) {
    // 7. Poll an SQS queue for its messages.
    std::vector<Aws::String> messages;
    std::vector<Aws::String> receiptHandles;
    while (true) {
        Aws::SQS::Model::ReceiveMessageRequest request;
        request.SetMaxNumberOfMessages(10);
        request.SetQueueUrl(queueURLS[i]);

        // Setting WaitTimeSeconds to non-zero enables long polling.
        // For information about long polling, see
        // https://docs.aws.amazon.com/AWSSimpleQueueService/latest/
        SQSDeveloperGuide/sqs-short-and-long-polling.html
    }
}
```

```
        request.SetWaitTimeSeconds(1);
        Aws::SQS::Model::ReceiveMessageOutcome outcome =
            sqsClient.ReceiveMessage(request);

        if (outcome.IsSuccess()) {
            const Aws::Vector<Aws::SQS::Model::Message> &newMessages =
outcome.GetResult().GetMessages();
            if (newMessages.empty()) {
                break;
            }
            else {
                for (const Aws::SQS::Model::Message &message: newMessages) {
                    messages.push_back(message.GetBody());
                    receiptHandles.push_back(message.GetReceiptHandle());
                }
            }
        }
        else {
            std::cerr << "Error with SQS::ReceiveMessage. "
                << outcome.GetError().GetMessage()
                << std::endl;

            cleanUp(topicARN,
                queueURLS,
                subscriptionARNS,
                snsClient,
                sqsClient);

            return false;
        }
    }

    printAsterisksLine();

    if (messages.empty()) {
        std::cout << "No messages were ";
    }
    else if (messages.size() == 1) {
        std::cout << "One message was ";
    }
    else {
        std::cout << messages.size() << " messages were ";
    }
    std::cout << "received by the queue '" << queueNames[i]
```

```
        << "'.'" << std::endl;
    for (const Aws::String &message: messages) {
        std::cout << " Message : '" << message << "'.'"
            << std::endl;
    }

    // 8. Delete a batch of messages from an SQS queue.
    if (!receiptHandles.empty()) {
        Aws::SQS::Model::DeleteMessageBatchRequest request;
        request.SetQueueUrl(queueURLS[i]);
        int id = 1; // Ids must be unique within a batch delete request.
        for (const Aws::String &receiptHandle: receiptHandles) {
            Aws::SQS::Model::DeleteMessageBatchRequestEntry entry;
            entry.SetId(std::to_string(id));
            ++id;
            entry.SetReceiptHandle(receiptHandle);
            request.AddEntries(entry);
        }

        Aws::SQS::Model::DeleteMessageBatchOutcome outcome =
            sqsClient.DeleteMessageBatch(request);

        if (outcome.IsSuccess()) {
            std::cout << "The batch deletion of messages was successful."
                << std::endl;
        }
        else {
            std::cerr << "Error with SQS::DeleteMessageBatch. "
                << outcome.GetError().GetMessage()
                << std::endl;
            cleanUp(topicARN,
                queueURLS,
                subscriptionARNS,
                snsClient,
                sqsClient);

            return false;
        }
    }
}

return cleanUp(topicARN,
    queueURLS,
    subscriptionARNS,
```

```

        snsClient,
        sqsClient,
        true); // askUser
    }

bool AwsDoc::TopicsAndQueues::cleanUp(const Aws::String &topicARN,
                                       const Aws::Vector<Aws::String> &queueURLS,
                                       const Aws::Vector<Aws::String>
                                       &subscriptionARNS,
                                       const Aws::SNS::SNSClient &snsClient,
                                       const Aws::SQS::SQSClient &sqsClient,
                                       bool askUser) {
    bool result = true;
    printAsterisksLine();
    if (!queueURLS.empty() && askUser &&
        askYesNoQuestion("Delete the SQS queues? (y/n) ")) {
        for (const auto &queueURL: queueURLS) {
            // 9. Delete an SQS queue.
            Aws::SQS::Model::DeleteQueueRequest request;
            request.SetQueueUrl(queueURL);

            Aws::SQS::Model::DeleteQueueOutcome outcome =
                sqsClient.DeleteQueue(request);

            if (outcome.IsSuccess()) {
                std::cout << "The queue with URL '" << queueURL
                    << "' was successfully deleted." << std::endl;
            }
            else {
                std::cerr << "Error with SQS::DeleteQueue. "
                    << outcome.GetError().GetMessage()
                    << std::endl;
                result = false;
            }
        }
    }

    for (const auto &subscriptionARN: subscriptionARNS) {
        // 10. Unsubscribe an SNS subscription.
        Aws::SNS::Model::UnsubscribeRequest request;
        request.SetSubscriptionArn(subscriptionARN);

        Aws::SNS::Model::UnsubscribeOutcome outcome =

```

```
        snsClient.Unsubscribe(request);

        if (outcome.IsSuccess()) {
            std::cout << "Unsubscribe of subscription ARN '" <<
subscriptionARN
                    << "' was successful." << std::endl;
        }
        else {
            std::cerr << "Error with TopicsAndQueues::Unsubscribe. "
                    << outcome.GetError().GetMessage()
                    << std::endl;
            result = false;
        }
    }
}

printAsterisksLine();
if (!topicARN.empty() && askUser &&
    askYesNoQuestion("Delete the SNS topic? (y/n) ")) {

    // 11. Delete an SNS topic.
    Aws::SNS::Model::DeleteTopicRequest request;
    request.SetTopicArn(topicARN);

    Aws::SNS::Model::DeleteTopicOutcome outcome =
snsClient.DeleteTopic(request);

    if (outcome.IsSuccess()) {
        std::cout << "The topic with ARN '" << topicARN
                << "' was successfully deleted." << std::endl;
    }
    else {
        std::cerr << "Error with TopicsAndQueues::DeleteTopicRequest. "
                << outcome.GetError().GetMessage()
                << std::endl;
        result = false;
    }
}

return result;
}

//! Create an IAM policy that gives an SQS queue permission to receive messages
from an SNS topic.
```

```

/ *!
 \sa createPolicyForQueue()
 \param queueARN: The SQS queue Amazon Resource Name (ARN).
 \param topicARN: The SNS topic ARN.
 \return Aws::String: The policy as JSON.
 */
Aws::String AwsDoc::TopicsAndQueues::createPolicyForQueue(const Aws::String
&queueARN,
                                                                    const Aws::String
&topicARN) {
    std::ostringstream policyStream;
    policyStream << R"({
        "Statement": [
            {
                "Effect": "Allow",
                "Principal": {
                    "Service": "sns.amazonaws.com"
                },
                "Action": "sqs:SendMessage",
                "Resource": ")" << queueARN << R"(",
                "Condition": {
                    "ArnEquals": {
                        "aws:SourceArn": ")" << topicARN << R"("
                    }
                }
            }
        ]
    })";

    return policyStream.str();
}

```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API AWS SDK for C++ .
  - [CreateQueue](#)
  - [CreateTopic](#)
  - [DeleteMessageBatch](#)
  - [DeleteQueue](#)
  - [DeleteTopic](#)
  - [GetQueueAttributes](#)

- [Publicar](#)
- [ReceiveMessage](#)
- [SetQueueAttributes](#)
- [Assinar](#)
- [Cancelar assinatura](#)

Go

SDK para Go V2

 Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

Execute um cenário interativo em um prompt de comando.

```
const FIFO_SUFFIX = ".fifo"
const TONE_KEY = "tone"

var ToneChoices = []string{"cheerful", "funny", "serious", "sincere"}

// MessageBody is used to deserialize the body of a message from a JSON string.
type MessageBody struct {
    Message string
}

// ScenarioRunner separates the steps of this scenario into individual functions
// so that
// they are simpler to read and understand.
type ScenarioRunner struct {
    questioner demotools.IQuestioner
    snsActor   *actions.SnsActions
    sqsActor   *actions.SqsActions
}

func (runner ScenarioRunner) CreateTopic() (string, string, bool, bool) {
```

```
log.Println("SNS topics can be configured as FIFO (First-In-First-Out) or
standard.\n" +
  "FIFO topics deliver messages in order and support deduplication and message
filtering.")
isFifoTopic := runner.questioner.AskBool("\nWould you like to work with FIFO
topics? (y/n) ", "y")

contentBasedDeduplication := false
if isFifoTopic {
  log.Println(strings.Repeat("-", 88))
  log.Println("Because you have chosen a FIFO topic, deduplication is supported.
\n" +
    "Deduplication IDs are either set in the message or are automatically
generated\n" +
    "from content using a hash function. If a message is successfully published to
\n" +
    "an SNS FIFO topic, any message published and determined to have the same\n" +
    "deduplication ID, within the five-minute deduplication interval, is accepted
\n" +
    "but not delivered. For more information about deduplication, see:\n" +
    "\thttps://docs.aws.amazon.com/sns/latest/dg/fifo-message-dedup.html.")
  contentBasedDeduplication = runner.questioner.AskBool(
    "\nDo you want to use content-based deduplication instead of entering a
deduplication ID? (y/n) ", "y")
}
log.Println(strings.Repeat("-", 88))

topicName := runner.questioner.Ask("Enter a name for your SNS topic. ")
if isFifoTopic {
  topicName = fmt.Sprintf("%v%v", topicName, FIFO_SUFFIX)
  log.Printf("Because you have selected a FIFO topic, '%v' must be appended to
\n"+
    "the topic name.", FIFO_SUFFIX)
}

topicArn, err := runner.snsActor.CreateTopic(topicName, isFifoTopic,
contentBasedDeduplication)
if err != nil {
  panic(err)
}
log.Printf("Your new topic with the name '%v' and Amazon Resource Name (ARN)
\n"+
  "'%v' has been created.", topicName, topicArn)
```

```

    return topicName, topicArn, isFifoTopic, contentBasedDeduplication
}

func (runner ScenarioRunner) CreateQueue(ordinal string, isFifoTopic bool)
(string, string) {
    queueName := runner.questioner.Ask(fmt.Sprintf("Enter a name for the %v SQS
queue. ", ordinal))
    if isFifoTopic {
        queueName = fmt.Sprintf("%v%v", queueName, FIFO_SUFFIX)
        if ordinal == "first" {
            log.Printf("Because you are creating a FIFO SQS queue, '%v' must "+
                "be appended to the queue name.\n", FIFO_SUFFIX)
        }
    }
    queueUrl, err := runner.sqsActor.CreateQueue(queueName, isFifoTopic)
    if err != nil {
        panic(err)
    }
    log.Printf("Your new SQS queue with the name '%v' and the queue URL "+
        "'%v' has been created.", queueName, queueUrl)

    return queueName, queueUrl
}

func (runner ScenarioRunner) SubscribeQueueToTopic(
    queueName string, queueUrl string, topicName string, topicArn string, ordinal
string,
    isFifoTopic bool) (string, bool) {

    queueArn, err := runner.sqsActor.GetQueueArn(queueUrl)
    if err != nil {
        panic(err)
    }
    log.Printf("The ARN of your queue is: %v.\n", queueArn)

    err = runner.sqsActor.AttachSendMessagePolicy(queueUrl, queueArn, topicArn)
    if err != nil {
        panic(err)
    }
    log.Println("Attached an IAM policy to the queue so the SNS topic can send " +
        "messages to it.")
    log.Println(strings.Repeat("-", 88))

    var filterPolicy map[string][]string

```

```
if isFifoTopic {
    if ordinal == "first" {
        log.Println("Subscriptions to a FIFO topic can have filters.\n" +
            "If you add a filter to this subscription, then only the filtered messages\n"
            +
            "will be received in the queue.\n" +
            "For information about message filtering, see\n" +
            "\thttps://docs.aws.amazon.com/sns/latest/dg/sns-message-filtering.html\n" +
            "For this example, you can filter messages by a \"tone\" attribute.")
    }

    wantFiltering := runner.questioner.AskBool(
        fmt.Sprintf("Do you want to filter messages that are sent to \"%v\"\n"+
            "from the %v topic? (y/n) ", queueName, topicName), "y")
    if wantFiltering {
        log.Println("You can filter messages by one or more of the following \"tone\"
            attributes.")

        var toneSelections []string
        askAboutTones := true
        for askAboutTones {
            toneIndex := runner.questioner.AskChoice(
                "Enter the number of the tone you want to filter by:\n", ToneChoices)
            toneSelections = append(toneSelections, ToneChoices[toneIndex])
            askAboutTones = runner.questioner.AskBool("Do you want to add another tone to
            the filter? (y/n) ", "y")
        }
        log.Printf("Your subscription will be filtered to only pass the following
            tones: %v\n", toneSelections)
        filterPolicy = map[string][]string{TONE_KEY: toneSelections}
    }
}

subscriptionArn, err := runner.snsActor.SubscribeQueue(topicArn, queueArn,
    filterPolicy)
if err != nil {
    panic(err)
}
log.Printf("The queue %v is now subscribed to the topic %v with the subscription
    ARN %v.\n",
    queueName, topicName, subscriptionArn)

return subscriptionArn, filterPolicy != nil
}
```

```
func (runner ScenarioRunner) PublishMessages(topicArn string, isFifoTopic bool,
contentBasedDeduplication bool, usingFilters bool) {
    var message string
    var groupId string
    var dedupId string
    var toneSelection string
    publishMore := true
    for publishMore {
        groupId = ""
        dedupId = ""
        toneSelection = ""
        message = runner.questioner.Ask("Enter a message to publish: ")
        if isFifoTopic {
            log.Println("Because you are using a FIFO topic, you must set a message group
ID.\n" +
                "All messages within the same group will be received in the order they were
published.")
            groupId = runner.questioner.Ask("Enter a message group ID: ")
            if !contentBasedDeduplication {
                log.Println("Because you are not using content-based deduplication,\n" +
                    "you must enter a deduplication ID.")
                dedupId = runner.questioner.Ask("Enter a deduplication ID: ")
            }
        }
        if usingFilters {
            if runner.questioner.AskBool("Add a tone attribute so this message can be
filtered? (y/n) ", "y") {
                toneIndex := runner.questioner.AskChoice(
                    "Enter the number of the tone you want to filter by:\n", ToneChoices)
                toneSelection = ToneChoices[toneIndex]
            }
        }

        err := runner.snsActor.Publish(topicArn, message, groupId, dedupId, TONE_KEY,
toneSelection)
        if err != nil {
            panic(err)
        }
        log.Println(("Your message was published.))

        publishMore = runner.questioner.AskBool("Do you want to publish another
message? (y/n) ", "y")
    }
}
```

```
}

func (runner ScenarioRunner) PollForMessages(queueUrls []string) {
    log.Println("Polling queues for messages...")
    for _, queueUrl := range queueUrls {
        var messages []types.Message
        for {
            currentMsgs, err := runner.sqsActor.GetMessages(queueUrl, 10, 1)
            if err != nil {
                panic(err)
            }
            if len(currentMsgs) == 0 {
                break
            }
            messages = append(messages, currentMsgs...)
        }
        if len(messages) == 0 {
            log.Printf("No messages were received by queue %v.\n", queueUrl)
        } else if len(messages) == 1 {
            log.Printf("One message was received by queue %v:\n", queueUrl)

        } else {
            log.Printf("%v messages were received by queue %v:\n", len(messages),
                queueUrl)
        }
        for msgIndex, message := range messages {
            messageBody := MessageBody{}
            err := json.Unmarshal([]byte(*message.Body), &messageBody)
            if err != nil {
                panic(err)
            }
            log.Printf("Message %v: %v\n", msgIndex+1, messageBody.Message)
        }

        if len(messages) > 0 {
            log.Printf("Deleting %v messages from queue %v.\n", len(messages), queueUrl)
            err := runner.sqsActor.DeleteMessages(queueUrl, messages)
            if err != nil {
                panic(err)
            }
        }
    }
}
```

```
// RunTopicsAndQueuesScenario is an interactive example that shows you how to use
// the
// AWS SDK for Go to create and use Amazon SNS topics and Amazon SQS queues.
//
// 1. Create a topic (FIFO or non-FIFO).
// 2. Subscribe several queues to the topic with an option to apply a filter.
// 3. Publish messages to the topic.
// 4. Poll the queues for messages received.
// 5. Delete the topic and the queues.
//
// This example creates service clients from the specified sdkConfig so that
// you can replace it with a mocked or stubbed config for unit testing.
//
// It uses a questioner from the `demotools` package to get input during the
// example.
// This package can be found in the ..\..\demotools folder of this repo.
func RunTopicsAndQueuesScenario(
    sdkConfig aws.Config, questioner demotools.IQuestioner) {
    resources := Resources{}
    defer func() {
        if r := recover(); r != nil {
            log.Println("Something went wrong with the demo.\n" +
                "Cleaning up any resources that were created...")
            resources.Cleanup()
        }
    }()
    queueCount := 2

    log.Println(strings.Repeat("-", 88))
    log.Printf("Welcome to messaging with topics and queues.\n\n"+
        "In this workflow, you will create an SNS topic and subscribe %v SQS queues to
        the\n"+
        "topic. You can select from several options for configuring the topic and the
        \n"+
        "subscriptions for the queues. You can then post to the topic and see the
        results\n"+
        "in the queues.\n", queueCount)

    log.Println(strings.Repeat("-", 88))

    runner := ScenarioRunner{
        questioner: questioner,
        snsActor:   &actions.SnsActions{SnsClient: sns.NewFromConfig(sdkConfig)},
        sqsActor:   &actions.SqsActions{SqsClient: sqs.NewFromConfig(sdkConfig)},
```

```
}
resources.snsActor = runner.snsActor
resources.sqsActor = runner.sqsActor

topicName, topicArn, isFifoTopic, contentBasedDeduplication :=
runner.CreateTopic()
resources.topicArn = topicArn
log.Println(strings.Repeat("-", 88))

log.Printf("Now you will create %v SQS queues and subscribe them to the topic.
\n", queueCount)
ordinals := []string{"first", "next"}
usingFilters := false
for _, ordinal := range ordinals {
    queueName, queueUrl := runner.CreateQueue(ordinal, isFifoTopic)
    resources.queueUrls = append(resources.queueUrls, queueUrl)

    _, filtering := runner.SubscribeQueueToTopic(queueName, queueUrl, topicName,
topicArn, ordinal, isFifoTopic)
    usingFilters = usingFilters || filtering
}

log.Println(strings.Repeat("-", 88))
runner.PublishMessages(topicArn, isFifoTopic, contentBasedDeduplication,
usingFilters)
log.Println(strings.Repeat("-", 88))
runner.PollForMessages(resources.queueUrls)

log.Println(strings.Repeat("-", 88))

wantCleanup := questioner.AskBool("Do you want to remove all AWS resources
created for this scenario? (y/n) ", "y")
if wantCleanup {
    log.Println("Cleaning up resources...")
    resources.Cleanup()
}

log.Println(strings.Repeat("-", 88))
log.Println("Thanks for watching!")
log.Println(strings.Repeat("-", 88))
}
```

Defina uma estrutura que envolva as ações do Amazon SNS usadas neste exemplo.

```
// SnsActions encapsulates the Amazon Simple Notification Service (Amazon SNS)
// actions
// used in the examples.
type SnsActions struct {
    SnsClient *sns.Client
}

// CreateTopic creates an Amazon SNS topic with the specified name. You can
// optionally
// specify that the topic is created as a FIFO topic and whether it uses content-
// based
// deduplication instead of ID-based deduplication.
func (actor SnsActions) CreateTopic(topicName string, isFifoTopic bool,
    contentBasedDeduplication bool) (string, error) {
    var topicArn string
    topicAttributes := map[string]string{}
    if isFifoTopic {
        topicAttributes["FifoTopic"] = "true"
    }
    if contentBasedDeduplication {
        topicAttributes["ContentBasedDeduplication"] = "true"
    }
    topic, err := actor.SnsClient.CreateTopic(context.TODO(), &sns.CreateTopicInput{
        Name:      aws.String(topicName),
        Attributes: topicAttributes,
    })
    if err != nil {
        log.Printf("Couldn't create topic %v. Here's why: %v\n", topicName, err)
    } else {
        topicArn = *topic.TopicArn
    }

    return topicArn, err
}

// DeleteTopic delete an Amazon SNS topic.
func (actor SnsActions) DeleteTopic(topicArn string) error {
```

```
_, err := actor.SnsClient.DeleteTopic(context.TODO(), &sns.DeleteTopicInput{
    TopicArn: aws.String(topicArn)})
if err != nil {
    log.Printf("Couldn't delete topic %v. Here's why: %v\n", topicArn, err)
}
return err
}

// SubscribeQueue subscribes an Amazon Simple Queue Service (Amazon SQS) queue to
// an
// Amazon SNS topic. When filterMap is not nil, it is used to specify a filter
// policy
// so that messages are only sent to the queue when the message has the specified
// attributes.
func (actor SnsActions) SubscribeQueue(topicArn string, queueArn string,
    filterMap map[string][]string) (string, error) {
    var subscriptionArn string
    var attributes map[string]string
    if filterMap != nil {
        filterBytes, err := json.Marshal(filterMap)
        if err != nil {
            log.Printf("Couldn't create filter policy, here's why: %v\n", err)
            return "", err
        }
        attributes = map[string]string{"FilterPolicy": string(filterBytes)}
    }
    output, err := actor.SnsClient.Subscribe(context.TODO(), &sns.SubscribeInput{
        Protocol:          aws.String("sqs"),
        TopicArn:          aws.String(topicArn),
        Attributes:         attributes,
        Endpoint:          aws.String(queueArn),
        ReturnSubscriptionArn: true,
    })
    if err != nil {
        log.Printf("Couldn't subscribe queue %v to topic %v. Here's why: %v\n",
            queueArn, topicArn, err)
    } else {
        subscriptionArn = *output.SubscriptionArn
    }

    return subscriptionArn, err
}
```

```
// Publish publishes a message to an Amazon SNS topic. The message is then sent
// to all
// subscribers. When the topic is a FIFO topic, the message must also contain a
// group ID
// and, when ID-based deduplication is used, a deduplication ID. An optional key-
// value
// filter attribute can be specified so that the message can be filtered
// according to
// a filter policy.
func (actor SnsActions) Publish(topicArn string, message string, groupId string,
dedupId string, filterKey string, filterValue string) error {
    publishInput := sns.PublishInput{TopicArn: aws.String(topicArn), Message:
aws.String(message)}
    if groupId != "" {
        publishInput.MessageGroupId = aws.String(groupId)
    }
    if dedupId != "" {
        publishInput.MessageDeduplicationId = aws.String(dedupId)
    }
    if filterKey != "" && filterValue != "" {
        publishInput.MessageAttributes = map[string]types.MessageAttributeValue{
            filterKey: {DataType: aws.String("String"), StringValue:
aws.String(filterValue)},
        }
    }
    _, err := actor.SnsClient.Publish(context.TODO(), &publishInput)
    if err != nil {
        log.Printf("Couldn't publish message to topic %v. Here's why: %v", topicArn,
err)
    }
    return err
}
```

Defina uma estrutura que envolva as ações do Amazon SQS usadas neste exemplo.

```
// SqsActions encapsulates the Amazon Simple Queue Service (Amazon SQS) actions
// used in the examples.
```

```
type SqsActions struct {
    SqsClient *sqs.Client
}

// CreateQueue creates an Amazon SQS queue with the specified name. You can
// specify
// whether the queue is created as a FIFO queue.
func (actor SqsActions) CreateQueue(queueName string, isFifoQueue bool) (string,
error) {
    var queueUrl string
    queueAttributes := map[string]string{}
    if isFifoQueue {
        queueAttributes["FifoQueue"] = "true"
    }
    queue, err := actor.SqsClient.CreateQueue(context.TODO(), &sqs.CreateQueueInput{
        QueueName: aws.String(queueName),
        Attributes: queueAttributes,
    })
    if err != nil {
        log.Printf("Couldn't create queue %v. Here's why: %v\n", queueName, err)
    } else {
        queueUrl = *queue.QueueUrl
    }

    return queueUrl, err
}

// GetQueueArn uses the GetQueueAttributes action to get the Amazon Resource Name
// (ARN)
// of an Amazon SQS queue.
func (actor SqsActions) GetQueueArn(queueUrl string) (string, error) {
    var queueArn string
    arnAttributeName := types.QueueAttributeNameQueueArn
    attribute, err := actor.SqsClient.GetQueueAttributes(context.TODO(),
&sqs.GetQueueAttributesInput{
        QueueUrl:      aws.String(queueUrl),
        AttributeNames: []types.QueueAttributeName{arnAttributeName},
    })
    if err != nil {
        log.Printf("Couldn't get ARN for queue %v. Here's why: %v\n", queueUrl, err)
    }
}
```

```
    } else {
        queueArn = attribute.Attributes[string(arnAttributeName)]
    }
    return queueArn, err
}

// AttachSendMessagePolicy uses the SetQueueAttributes action to attach a policy
// to an
// Amazon SQS queue that allows the specified Amazon SNS topic to send messages
// to the
// queue.
func (actor SqsActions) AttachSendMessagePolicy(queueUrl string, queueArn string,
    topicArn string) error {
    policyDoc := PolicyDocument{
        Version: "2012-10-17",
        Statement: []PolicyStatement{{
            Effect:    "Allow",
            Action:   "sqs:SendMessage",
            Principal: map[string]string{"Service": "sns.amazonaws.com"},
            Resource:  aws.String(queueArn),
            Condition: PolicyCondition{"ArnEquals": map[string]string{"aws:SourceArn":
                topicArn}},
        }},
    }
    policyBytes, err := json.Marshal(policyDoc)
    if err != nil {
        log.Printf("Couldn't create policy document. Here's why: %v\n", err)
        return err
    }
    _, err = actor.SqsClient.SetQueueAttributes(context.TODO(),
        &sqs.SetQueueAttributesInput{
            Attributes: map[string]string{
                string(types.QueueAttributeNamePolicy): string(policyBytes),
            },
            QueueUrl: aws.String(queueUrl),
        })
    if err != nil {
        log.Printf("Couldn't set send message policy on queue %v. Here's why: %v\n",
            queueUrl, err)
    }
    return err
}
```

```
// PolicyDocument defines a policy document as a Go struct that can be serialized
// to JSON.
type PolicyDocument struct {
    Version    string
    Statement []PolicyStatement
}

// PolicyStatement defines a statement in a policy document.
type PolicyStatement struct {
    Effect      string
    Action     string
    Principal  map[string]string `json:",omitempty"`
    Resource   *string            `json:",omitempty"`
    Condition  PolicyCondition   `json:",omitempty"`
}

// PolicyCondition defines a condition in a policy.
type PolicyCondition map[string]map[string]string

// GetMessages uses the ReceiveMessage action to get messages from an Amazon SQS
// queue.
func (actor SqsActions) GetMessages(queueUrl string, maxMessages int32, waitTime
int32) ([]types.Message, error) {
    var messages []types.Message
    result, err := actor.SqsClient.ReceiveMessage(context.TODO(),
&sqs.ReceiveMessageInput{
        QueueUrl:          aws.String(queueUrl),
        MaxNumberOfMessages: maxMessages,
        WaitTimeSeconds:   waitTime,
    })
    if err != nil {
        log.Printf("Couldn't get messages from queue %v. Here's why: %v\n", queueUrl,
err)
    } else {
        messages = result.Messages
    }
    return messages, err
}
```

```
// DeleteMessages uses the DeleteMessageBatch action to delete a batch of
// messages from
// an Amazon SQS queue.
func (actor SqsActions) DeleteMessages(queueUrl string, messages []types.Message)
error {
    entries := make([]types.DeleteMessageBatchRequestEntry, len(messages))
    for msgIndex := range messages {
        entries[msgIndex].Id = aws.String(fmt.Sprintf("%v", msgIndex))
        entries[msgIndex].ReceiptHandle = messages[msgIndex].ReceiptHandle
    }
    _, err := actor.SqsClient.DeleteMessageBatch(context.TODO(),
    &sqs.DeleteMessageBatchInput{
        Entries: entries,
        QueueUrl: aws.String(queueUrl),
    })
    if err != nil {
        log.Printf("Couldn't delete messages from queue %v. Here's why: %v\n",
        queueUrl, err)
    }
    return err
}

// DeleteQueue deletes an Amazon SQS queue.
func (actor SqsActions) DeleteQueue(queueUrl string) error {
    _, err := actor.SqsClient.DeleteQueue(context.TODO(), &sqs.DeleteQueueInput{
        QueueUrl: aws.String(queueUrl)})
    if err != nil {
        log.Printf("Couldn't delete queue %v. Here's why: %v\n", queueUrl, err)
    }
    return err
}
```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API AWS SDK for Go .
  - [CreateQueue](#)
  - [CreateTopic](#)
  - [DeleteMessageBatch](#)
  - [DeleteQueue](#)

- [DeleteTopic](#)
- [GetQueueAttributes](#)
- [Publicar](#)
- [ReceiveMessage](#)
- [SetQueueAttributes](#)
- [Assinar](#)
- [Cancelar assinatura](#)

## JavaScript

### SDK para JavaScript (v3)

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

Esse é o ponto de entrada para esse fluxo de trabalho.

```
import { SNSClient } from "@aws-sdk/client-sns";
import { SQSClient } from "@aws-sdk/client-sqs";

import { TopicsQueuesWkflw } from "./TopicsQueuesWkflw.js";
import { Prompter } from "@aws-doc-sdk-examples/lib/prompter.js";
import { SlowLogger } from "@aws-doc-sdk-examples/lib/slow-logger.js";

export const startSnsWorkflow = () => {
  const noLoggerDelay = process.argv.find((arg) => arg === "--no-logger-delay");
  const snsClient = new SNSClient({});
  const sqsClient = new SQSClient({});
  const prompter = new Prompter();
  const logger = noLoggerDelay ? console : new SlowLogger(25);

  const wkflw = new TopicsQueuesWkflw(snsClient, sqsClient, prompter, logger);

  wkflw.start();
};
```

O código anterior fornece as dependências necessárias e inicia o fluxo de trabalho. A próxima seção contém a maior parte do exemplo.

```
const toneChoices = [
  { name: "cheerful", value: "cheerful" },
  { name: "funny", value: "funny" },
  { name: "serious", value: "serious" },
  { name: "sincere", value: "sincere" },
];

export class TopicsQueuesWkflw {
  // SNS topic is configured as First-In-First-Out
  isFifo = true;

  // Automatic content-based deduplication is enabled.
  autoDedup = false;

  snsClient;
  sqsClient;
  topicName;
  topicArn;
  subscriptionArns = [];
  /**
   * @type {{ queueName: string, queueArn: string, queueUrl: string, policy?:
string }[]}
   */
  queues = [];
  prompter;

  /**
   * @param {import('@aws-sdk/client-sns').SNSClient} snsClient
   * @param {import('@aws-sdk/client-sqs').SQSClient} sqsClient
   * @param {import('../libs/prompter.js').Prompter} prompter
   * @param {import('../libs/logger.js').Logger} logger
   */
  constructor(snsClient, sqsClient, prompter, logger) {
    this.snsClient = snsClient;
    this.sqsClient = sqsClient;
    this.prompter = prompter;
    this.logger = logger;
  }
}
```

```
}

async welcome() {
  await this.logger.log(MESSAGES.description);
}

async confirmFifo() {
  await this.logger.log(MESSAGES.snsFifoDescription);
  this.isFifo = await this.prompter.confirm({
    message: MESSAGES.snsFifoPrompt,
  });

  if (this.isFifo) {
    this.logger.logSeparator(MESSAGES.headerDedup);
    await this.logger.log(MESSAGES.deduplicationNotice);
    await this.logger.log(MESSAGES.deduplicationDescription);
    this.autoDedup = await this.prompter.confirm({
      message: MESSAGES.deduplicationPrompt,
    });
  }
}

async createTopic() {
  await this.logger.log(MESSAGES.creatingTopics);
  this.topicName = await this.prompter.input({
    message: MESSAGES.topicNamePrompt,
  });
  if (this.isFifo) {
    this.topicName += ".fifo";
    this.logger.logSeparator(MESSAGES.headerFifoNaming);
    await this.logger.log(MESSAGES.appendFifoNotice);
  }

  const response = await this.snsClient.send(
    new CreateTopicCommand({
      Name: this.topicName,
      Attributes: {
        FifoTopic: this.isFifo ? "true" : "false",
        ...(this.autoDedup ? { ContentBasedDeduplication: "true" } : {}),
      },
    }),
  );

  this.topicArn = response.TopicArn;
```

```
await this.logger.log(
  MESSAGES.topicCreatedNotice
    .replace("${TOPIC_NAME}", this.topicName)
    .replace("${TOPIC_ARN}", this.topicArn),
);
}

async createQueues() {
  await this.logger.log(MESSAGES.createQueuesNotice);
  // Increase this number to add more queues.
  let maxQueues = 2;

  for (let i = 0; i < maxQueues; i++) {
    await this.logger.log(MESSAGES.queueCount.replace("${COUNT}", i + 1));
    let queueName = await this.prompter.input({
      message: MESSAGES.queueNamePrompt.replace(
        "${EXAMPLE_NAME}",
        i === 0 ? "good-news" : "bad-news",
      ),
    });

    if (this.isFifo) {
      queueName += ".fifo";
      await this.logger.log(MESSAGES.appendFifoNotice);
    }

    const response = await this.sqsClient.send(
      new CreateQueueCommand({
        QueueName: queueName,
        Attributes: { ...(this.isFifo ? { FifoQueue: "true" } : {}) },
      }),
    );

    const { Attributes } = await this.sqsClient.send(
      new GetQueueAttributesCommand({
        QueueUrl: response.QueueUrl,
        AttributeNames: ["QueueArn"],
      }),
    );

    this.queues.push({
      queueName,
      queueArn: Attributes.QueueArn,
    });
  }
}
```

```
        queueUrl: response.QueueUrl,
    });

    await this.logger.log(
        MESSAGES.queueCreatedNotice
            .replace("${QUEUE_NAME}", queueName)
            .replace("${QUEUE_URL}", response.QueueUrl)
            .replace("${QUEUE_ARN}", Attributes.QueueArn),
    );
}
}

async attachQueueIamPolicies() {
    for (const [index, queue] of this.queues.entries()) {
        const policy = JSON.stringify(
            {
                Statement: [
                    {
                        Effect: "Allow",
                        Principal: {
                            Service: "sns.amazonaws.com",
                        },
                        Action: "sqs:SendMessage",
                        Resource: queue.queueArn,
                        Condition: {
                            ArnEquals: {
                                "aws:SourceArn": this.topicArn,
                            },
                        },
                    },
                ],
            },
            null,
            2,
        );

        if (index !== 0) {
            this.logger.logSeparator();
        }

        await this.logger.log(MESSAGES.attachPolicyNotice);
        console.log(policy);
        const addPolicy = await this.prompter.confirm({
            message: MESSAGES.addPolicyConfirmation.replace(
```

```
        "${QUEUE_NAME}",
        queue.queueName,
    ),
});

if (addPolicy) {
    await this.sqsClient.send(
        new SetQueueAttributesCommand({
            QueueUrl: queue.queueUrl,
            Attributes: {
                Policy: policy,
            },
        }),
    );
    queue.policy = policy;
} else {
    await this.logger.log(
        MESSAGES.policyNotAttachedNotice.replace(
            "${QUEUE_NAME}",
            queue.queueName,
        ),
    );
}
}
}

async subscribeQueuesToTopic() {
    for (const [index, queue] of this.queues.entries()) {
        /**
         * @type {import('@aws-sdk/client-sns').SubscribeCommandInput}
         */
        const subscribeParams = {
            TopicArn: this.topicArn,
            Protocol: "sqs",
            Endpoint: queue.queueArn,
        };
        let tones = [];

        if (this.isFifo) {
            if (index === 0) {
                await this.logger.log(MESSAGES.fifoFilterNotice);
            }
            tones = await this.prompter.checkbox({
                message: MESSAGES.fifoFilterSelect.replace(
```

```
        "${QUEUE_NAME}",
        queue.queueName,
    ),
    choices: toneChoices,
});

if (tones.length) {
    subscribeParams.Attributes = {
        FilterPolicyScope: "MessageAttributes",
        FilterPolicy: JSON.stringify({
            tone: tones,
        }),
    };
}

const { SubscriptionArn } = await this.snsClient.send(
    new SubscribeCommand(subscribeParams),
);

this.subscriptionArns.push(SubscriptionArn);

await this.logger.log(
    MESSAGES.queueSubscribedNotice
        .replace("${QUEUE_NAME}", queue.queueName)
        .replace("${TOPIC_NAME}", this.topicName)
        .replace("${TONES}", tones.length ? tones.join(", ") : "none"),
);
}
}

async publishMessages() {
    const message = await this.prompter.input({
        message: MESSAGES.publishMessagePrompt,
    });

    let groupId, deduplicationId, choices;

    if (this.isFifo) {
        await this.logger.log(MESSAGES.groupIdNotice);
        groupId = await this.prompter.input({
            message: MESSAGES.groupIdPrompt,
        });
    }
}
```

```
if (this.autoDedup === false) {
  await this.logger.log(MESSAGES.deduplicationIdNotice);
  deduplicationId = await this.prompter.input({
    message: MESSAGES.deduplicationIdPrompt,
  });
}

choices = await this.prompter.checkbox({
  message: MESSAGES.messageAttributesPrompt,
  choices: toneChoices,
});
}

await this.snsClient.send(
  new PublishCommand({
    TopicArn: this.topicArn,
    Message: message,
    ...(groupId
      ? {
          MessageGroupId: groupId,
        }
      : {}),
    ...(deduplicationId
      ? {
          MessageDeduplicationId: deduplicationId,
        }
      : {}),
    ...(choices
      ? {
          MessageAttributes: {
            tone: {
              DataType: "String.Array",
              StringValue: JSON.stringify(choices),
            },
          },
        }
      : {}),
  })),
);

const publishAnother = await this.prompter.confirm({
  message: MESSAGES.publishAnother,
});
```

```
    if (publishAnother) {
      await this.publishMessages();
    }
  }

  async receiveAndDeleteMessages() {
    for (const queue of this.queues) {
      const { Messages } = await this.sqsClient.send(
        new ReceiveMessageCommand({
          QueueUrl: queue.queueUrl,
        }),
      );

      if (Messages) {
        await this.logger.log(
          MESSAGES.messagesReceivedNotice.replace(
            "${QUEUE_NAME}",
            queue.queueName,
          ),
        );
        console.log(Messages);

        await this.sqsClient.send(
          new DeleteMessageBatchCommand({
            QueueUrl: queue.queueUrl,
            Entries: Messages.map((message) => ({
              Id: message.MessageId,
              ReceiptHandle: message.ReceiptHandle,
            })),
          }),
        );
      } else {
        await this.logger.log(
          MESSAGES.noMessagesReceivedNotice.replace(
            "${QUEUE_NAME}",
            queue.queueName,
          ),
        );
      }
    }
  }

  const deleteAndPoll = await this.prompter.confirm({
    message: MESSAGES.deleteAndPollConfirmation,
  });
```

```
    if (deleteAndPoll) {
      await this.receiveAndDeleteMessages();
    }
  }

  async destroyResources() {
    for (const subscriptionArn of this.subscriptionArns) {
      await this.snsClient.send(
        new UnsubscribeCommand({ SubscriptionArn: subscriptionArn }),
      );
    }

    for (const queue of this.queues) {
      await this.sqsClient.send(
        new DeleteQueueCommand({ QueueUrl: queue.queueUrl }),
      );
    }

    if (this.topicArn) {
      await this.snsClient.send(
        new DeleteTopicCommand({ TopicArn: this.topicArn }),
      );
    }
  }

  async start() {
    console.clear();

    try {
      this.logger.logSeparator(MESSAGES.headerWelcome);
      await this.welcome();
      this.logger.logSeparator(MESSAGES.headerFifo);
      await this.confirmFifo();
      this.logger.logSeparator(MESSAGES.headerCreateTopic);
      await this.createTopic();
      this.logger.logSeparator(MESSAGES.headerCreateQueues);
      await this.createQueues();
      this.logger.logSeparator(MESSAGES.headerAttachPolicy);
      await this.attachQueueIamPolicies();
      this.logger.logSeparator(MESSAGES.headerSubscribeQueues);
      await this.subscribeQueuesToTopic();
      this.logger.logSeparator(MESSAGES.headerPublishMessage);
      await this.publishMessages();
    }
  }
}
```

```
        this.logger.logSeparator(MESSAGES.headerReceiveMessages);
        await this.receiveAndDeleteMessages();
    } catch (err) {
        console.error(err);
    } finally {
        await this.destroyResources();
    }
}
}
```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API AWS SDK for JavaScript .
  - [CreateQueue](#)
  - [CreateTopic](#)
  - [DeleteMessageBatch](#)
  - [DeleteQueue](#)
  - [DeleteTopic](#)
  - [GetQueueAttributes](#)
  - [Publicar](#)
  - [ReceiveMessage](#)
  - [SetQueueAttributes](#)
  - [Assinar](#)
  - [Cancelar assinatura](#)

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulte [Usando o Amazon SNS com um SDK AWS](#). Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

## Exemplos sem servidor para Amazon SNS usando SDKs AWS

Os exemplos de código a seguir mostram como usar o Amazon SNS com AWS SDKs.

### Exemplos

- [Invocar uma função do Lambda em um acionador do Amazon SNS](#)

## Invocar uma função do Lambda em um acionador do Amazon SNS

Os exemplos de código a seguir mostram como implementar uma função do Lambda que recebe um evento acionado pelo recebimento de mensagens de um tópico do SNS. A função recupera as mensagens do parâmetro event e registra o conteúdo de cada mensagem.

.NET

AWS SDK for .NET

### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no repositório dos [Exemplos sem servidor](#).

Consumir um evento do SNS com o Lambda usando .NET.

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0
using Amazon.Lambda.Core;
using Amazon.Lambda.SNSEvents;

// Assembly attribute to enable the Lambda function's JSON input to be converted
// into a .NET class.
[assembly:
    LambdaSerializer(typeof(Amazon.Lambda.Serialization.SystemTextJson.DefaultLambdaJsonSerializer))]

namespace SnsIntegration;

public class Function
{
    public async Task FunctionHandler(SNSEvent evnt, ILambdaContext context)
    {
        foreach (var record in evnt.Records)
        {
            await ProcessRecordAsync(record, context);
        }
        context.Logger.LogInformation("done");
    }
}
```

```
private async Task ProcessRecordAsync(SNSEvent.SNSRecord record,
ILambdaContext context)
{
    try
    {
        context.Logger.LogInformation($"Processed record
{record.Sns.Message}");

        // TODO: Do interesting work based on the new message
        await Task.CompletedTask;
    }
    catch (Exception e)
    {
        //You can use Dead Letter Queue to handle failures. By configuring a
        Lambda DLQ.
        context.Logger.LogError($"An error occurred");
        throw;
    }
}
}
```

Go

## SDK para Go V2

### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no repositório dos [Exemplos sem servidor](#).

Consumir um evento do SNS com o Lambda usando Go.

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0
package main

import (
    "context"
    "fmt"

    "github.com/aws/aws-lambda-go/events"
```

```
"github.com/aws/aws-lambda-go/lambda"
)

func handler(ctx context.Context, snsEvent events.SNSEvent) {
    for _, record := range snsEvent.Records {
        processMessage(record)
    }
    fmt.Println("done")
}

func processMessage(record events.SNSEventRecord) {
    message := record.SNS.Message
    fmt.Printf("Processed message: %s\n", message)
    // TODO: Process your record here
}

func main() {
    lambda.Start(handler)
}
```

## Java

### SDK para Java 2.x

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no repositório dos [Exemplos sem servidor](#).

Consumir um evento do SNS com o Lambda usando Java.

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0
package example;

import com.amazonaws.services.lambda.runtime.Context;
import com.amazonaws.services.lambda.runtime.LambdaLogger;
import com.amazonaws.services.lambda.runtime.RequestHandler;
import com.amazonaws.services.lambda.runtime.events.SNSEvent;
import com.amazonaws.services.lambda.runtime.events.SNSEvent.SNSRecord;
```

```
import java.util.Iterator;
import java.util.List;

public class SNSEventHandler implements RequestHandler<SNSEvent, Boolean> {
    LambdaLogger logger;

    @Override
    public Boolean handleRequest(SNSEvent event, Context context) {
        logger = context.getLogger();
        List<SNSRecord> records = event.getRecords();
        if (!records.isEmpty()) {
            Iterator<SNSRecord> recordsIter = records.iterator();
            while (recordsIter.hasNext()) {
                processRecord(recordsIter.next());
            }
        }
        return Boolean.TRUE;
    }

    public void processRecord(SNSRecord record) {
        try {
            String message = record.getSNS().getMessage();
            logger.log("message: " + message);
        } catch (Exception e) {
            throw new RuntimeException(e);
        }
    }
}
```

## JavaScript

### SDK para JavaScript (v3)

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no repositório dos [Exemplos sem servidor](#).

Consumindo um evento do SNS com o JavaScript Lambda usando.

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0
exports.handler = async (event, context) => {
  for (const record of event.Records) {
    await processMessageAsync(record);
  }
  console.info("done");
};

async function processMessageAsync(record) {
  try {
    const message = JSON.stringify(record.Sns.Message);
    console.log(`Processed message ${message}`);
    await Promise.resolve(1); //Placeholder for actual async work
  } catch (err) {
    console.error("An error occurred");
    throw err;
  }
}
```

Consumindo um evento do SNS com o TypeScript Lambda usando.

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0
import { SNSEvent, Context, SNSHandler, SNSEventRecord } from "aws-lambda";

export const functionHandler: SNSHandler = async (
  event: SNSEvent,
  context: Context
```

```
    ): Promise<void> => {
      for (const record of event.Records) {
        await processMessageAsync(record);
      }
      console.info("done");
    };

    async function processMessageAsync(record: SNSEventRecord): Promise<any> {
      try {
        const message: string = JSON.stringify(record.Sns.Message);
        console.log(`Processed message ${message}`);
        await Promise.resolve(1); //Placeholder for actual async work
      } catch (err) {
        console.error("An error occurred");
        throw err;
      }
    }
  }
}
```

## PHP

### SDK para PHP

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no repositório dos [Exemplos sem servidor](#).

Consumir um evento do SNS com o Lambda usando PHP.

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0
<?php

/*
Since native PHP support for AWS Lambda is not available, we are utilizing Bref's
PHP functions runtime for AWS Lambda.
For more information on Bref's PHP runtime for Lambda, refer to: https://bref.sh/
docs/runtimes/function

Another approach would be to create a custom runtime.
```

```
A practical example can be found here: https://aws.amazon.com/blogs/apn/aws-lambda-custom-runtime-for-php-a-practical-example/
*/

// Additional composer packages may be required when using Bref or any other PHP
// functions runtime.
// require __DIR__ . '/vendor/autoload.php';

use Bref\Context\Context;
use Bref\Event\Sns\SnsEvent;
use Bref\Event\Sns\SnsHandler;

class Handler extends SnsHandler
{
    public function handleSns(SnsEvent $event, Context $context): void
    {
        foreach ($event->getRecords() as $record) {
            $message = $record->getMessage();

            // TODO: Implement your custom processing logic here
            // Any exception thrown will be logged and the invocation will be
            // marked as failed

            echo "Processed Message: $message" . PHP_EOL;
        }
    }
}

return new Handler();
```

## Python

### SDK para Python (Boto3)

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no repositório dos [Exemplos sem servidor](#).

Consumir um evento do SNS com o Lambda usando Python.

```
# Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
# SPDX-License-Identifier: Apache-2.0
def lambda_handler(event, context):
    for record in event['Records']:
        process_message(record)
    print("done")

def process_message(record):
    try:
        message = record['Sns']['Message']
        print(f"Processed message {message}")
        # TODO; Process your record here

    except Exception as e:
        print("An error occurred")
        raise e
```

## Ruby

### SDK para Ruby

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no repositório dos [Exemplos sem servidor](#).

Consumir um evento do SNS com o Lambda usando Ruby.

```
# Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
# SPDX-License-Identifier: Apache-2.0
def lambda_handler(event:, context:)
    event['Records'].map { |record| process_message(record) }
end

def process_message(record)
    message = record['Sns']['Message']
    puts("Processing message: #{message}")
rescue StandardError => e
    puts("Error processing message: #{e}")
```

```
    raise
end
```

## Rust

### SDK para Rust

#### Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no repositório dos [Exemplos sem servidor](#).

Consuma um evento do SNS com o Lambda usando Rust.

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0
use aws_lambda_events::event::sns::SnsEvent;
use aws_lambda_events::sns::SnsRecord;
use lambda_runtime::{run, service_fn, Error, LambdaEvent};
use tracing::info;

// Built with the following dependencies:
// aws_lambda_events = { version = "0.10.0", default-features = false, features
//   = ["sns"] }
// lambda_runtime = "0.8.1"
// tokio = { version = "1", features = ["macros"] }
// tracing = { version = "0.1", features = ["log"] }
// tracing-subscriber = { version = "0.3", default-features = false, features =
//   ["fmt"] }

async fn function_handler(event: LambdaEvent<SnsEvent>) -> Result<(), Error> {
    for event in event.payload.records {
        process_record(&event)?;
    }

    Ok(())
}

fn process_record(record: &SnsRecord) -> Result<(), Error> {
    info!("Processing SNS Message: {}", record.sns.message);
}
```

```
// Implement your record handling code here.

Ok(())
}

#[tokio::main]
async fn main() -> Result<(), Error> {
    tracing_subscriber::fmt()
        .with_max_level(tracing::Level::INFO)
        .with_target(false)
        .without_time()
        .init();

    run(service_fn(function_handler)).await
}
```

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulte [Usando o Amazon SNS com um SDK AWS](#). Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

## Exemplos de serviços cruzados para o Amazon SNS usando SDKs AWS

Os exemplos de aplicativos a seguir usam AWS SDKs para combinar o Amazon SNS com outros. Serviços da AWS Cada exemplo inclui um link para GitHub, onde você pode encontrar instruções sobre como configurar e executar o aplicativo.

### Exemplos

- [Criar uma aplicação para enviar dados para uma tabela do DynamoDB](#)
- [Criar uma aplicação de publicação e assinatura que traduz mensagens](#)
- [Criar uma aplicação de gerenciamento de ativos de fotos que permita que os usuários gerenciem fotos usando rótulos](#)
- [Criar uma aplicação de exploração do Amazon Textract](#)
- [Detecte pessoas e objetos em um vídeo com o Amazon Rekognition usando um SDK AWS](#)
- [Publique mensagens do Amazon SNS nas filas do Amazon SQS usando um SDK AWS](#)
- [Usar o API Gateway para invocar uma função do Lambda](#)

- [Usar eventos programados para invocar uma função do Lambda](#)

## Criar uma aplicação para enviar dados para uma tabela do DynamoDB

Os exemplos de código a seguir mostram como criar uma aplicação que envia dados para uma tabela do Amazon DynamoDB e notifica você quando um usuário atualiza a tabela.

### Java

#### SDK para Java 2.x

Mostra como criar uma aplicação Web dinâmica que envia dados usando a API Java do Amazon DynamoDB e envia uma mensagem de texto usando a API Java do Amazon Simple Notification Service.

Para obter o código-fonte completo e instruções sobre como configurar e executar, veja o exemplo completo em [GitHub](#).

Serviços usados neste exemplo

- DynamoDB
- Amazon SNS

### JavaScript

#### SDK para JavaScript (v3)

Este exemplo mostra como criar uma aplicação que permite que os usuários enviem dados para uma tabela do Amazon DynamoDB e enviem uma mensagem de texto ao administrador usando o Amazon Simple Notification Service (Amazon SNS).

Para obter o código-fonte completo e instruções sobre como configurar e executar, veja o exemplo completo em [GitHub](#).

Esse exemplo também está disponível no [Guia do desenvolvedor do AWS SDK for JavaScript v3](#).

Serviços usados neste exemplo

- DynamoDB

- Amazon SNS

## Kotlin

### SDK para Kotlin

Mostra como criar uma aplicação Android nativa que envia dados usando a API Kotlin do Amazon DynamoDB e envia uma mensagem de texto usando a API Kotlin do Amazon SNS.

Para obter o código-fonte completo e instruções sobre como configurar e executar, veja o exemplo completo em [GitHub](#).

Serviços usados neste exemplo

- DynamoDB
- Amazon SNS

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulte [Usando o Amazon SNS com um SDK AWS](#). Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

## Criar uma aplicação de publicação e assinatura que traduz mensagens

Os exemplos de código a seguir mostram como criar uma aplicação que oferece funcionalidade de assinatura e publicação e tradução de mensagens.

## .NET

### AWS SDK for .NET

Mostra como usar a API .NET do Amazon Simple Notification Service para criar uma aplicação Web com funcionalidade de assinatura e publicação. Além disso, essa aplicação de exemplo também traduz mensagens.

Para obter o código-fonte completo e instruções sobre como configurar e executar, veja o exemplo completo em [GitHub](#).

Serviços utilizados neste exemplo

- Amazon SNS
- Amazon Translate

## Java

### SDK para Java 2.x

Mostra como usar a API Java do Amazon Simple Notification Service para criar uma aplicação Web com funcionalidade de assinatura e publicação. Além disso, essa aplicação de exemplo também traduz mensagens.

Para obter o código-fonte completo e instruções sobre como configurar e executar, veja o exemplo completo em [GitHub](#).

Para obter o código-fonte completo e instruções sobre como configurar e executar o exemplo que usa a API Java Async, consulte o exemplo completo em [GitHub](#).

Serviços utilizados neste exemplo

- Amazon SNS
- Amazon Translate

## Kotlin

### SDK para Kotlin

Mostra como usar a API Kotlin do Amazon SNS para criar uma aplicação com funcionalidade de assinatura e publicação. Além disso, essa aplicação de exemplo também traduz mensagens.

Para obter o código-fonte completo e as instruções sobre como criar um aplicativo web, veja o exemplo completo em [GitHub](#).

Para ver o código-fonte completo e instruções sobre como criar um aplicativo Android nativo, veja o exemplo completo em [GitHub](#).

Serviços utilizados neste exemplo

- Amazon SNS
- Amazon Translate

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulte [Usando o Amazon SNS com um SDK AWS](#). Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

# Criar uma aplicação de gerenciamento de ativos de fotos que permita que os usuários gerenciem fotos usando rótulos

O exemplo de código a seguir mostra como criar uma aplicação com tecnologia sem servidor que permite que os usuários gerenciem fotos usando rótulos.

## .NET

### AWS SDK for .NET

Mostra como desenvolver uma aplicação de gerenciamento de ativos fotográficos que detecta rótulos em imagens usando o Amazon Rekognition e os armazena para recuperação posterior.

Para obter o código-fonte completo e instruções sobre como configurar e executar, veja o exemplo completo em [GitHub](#).

Para uma análise detalhada da origem desse exemplo, veja a publicação na [Comunidade da AWS](#).

Serviços utilizados neste exemplo

- API Gateway
- DynamoDB
- Lambda
- Amazon Rekognition
- Amazon S3
- Amazon SNS

## C++

### SDK para C++

Mostra como desenvolver uma aplicação de gerenciamento de ativos fotográficos que detecta rótulos em imagens usando o Amazon Rekognition e os armazena para recuperação posterior.

Para obter o código-fonte completo e instruções sobre como configurar e executar, veja o exemplo completo em [GitHub](#).

Para uma análise detalhada da origem desse exemplo, veja a publicação na [Comunidade da AWS](#).

Serviços utilizados neste exemplo

- API Gateway
- DynamoDB
- Lambda
- Amazon Rekognition
- Amazon S3
- Amazon SNS

## Java

### SDK para Java 2.x

Mostra como desenvolver uma aplicação de gerenciamento de ativos fotográficos que detecta rótulos em imagens usando o Amazon Rekognition e os armazena para recuperação posterior.

Para obter o código-fonte completo e instruções sobre como configurar e executar, veja o exemplo completo em [GitHub](#).

Para uma análise detalhada da origem desse exemplo, veja a publicação na [Comunidade da AWS](#).

Serviços utilizados neste exemplo

- API Gateway
- DynamoDB
- Lambda
- Amazon Rekognition
- Amazon S3
- Amazon SNS

## JavaScript

### SDK para JavaScript (v3)

Mostra como desenvolver uma aplicação de gerenciamento de ativos fotográficos que detecta rótulos em imagens usando o Amazon Rekognition e os armazena para recuperação posterior.

Para obter o código-fonte completo e instruções sobre como configurar e executar, veja o exemplo completo em [GitHub](#).

Para uma análise detalhada da origem desse exemplo, veja a publicação na [Comunidade da AWS](#).

Serviços utilizados neste exemplo

- API Gateway
- DynamoDB
- Lambda
- Amazon Rekognition
- Amazon S3
- Amazon SNS

## Kotlin

### SDK para Kotlin

Mostra como desenvolver uma aplicação de gerenciamento de ativos fotográficos que detecta rótulos em imagens usando o Amazon Rekognition e os armazena para recuperação posterior.

Para obter o código-fonte completo e instruções sobre como configurar e executar, veja o exemplo completo em [GitHub](#).

Para uma análise detalhada da origem desse exemplo, veja a publicação na [Comunidade da AWS](#).

Serviços utilizados neste exemplo

- API Gateway
- DynamoDB

- Lambda
- Amazon Rekognition
- Amazon S3
- Amazon SNS

## PHP

### SDK para PHP

Mostra como desenvolver uma aplicação de gerenciamento de ativos fotográficos que detecta rótulos em imagens usando o Amazon Rekognition e os armazena para recuperação posterior.

Para obter o código-fonte completo e instruções sobre como configurar e executar, veja o exemplo completo em [GitHub](#).

Para uma análise detalhada da origem desse exemplo, veja a publicação na [Comunidade da AWS](#).

Serviços utilizados neste exemplo

- API Gateway
- DynamoDB
- Lambda
- Amazon Rekognition
- Amazon S3
- Amazon SNS

## Rust

### SDK para Rust

Mostra como desenvolver uma aplicação de gerenciamento de ativos fotográficos que detecta rótulos em imagens usando o Amazon Rekognition e os armazena para recuperação posterior.

Para obter o código-fonte completo e instruções sobre como configurar e executar, veja o exemplo completo em [GitHub](#).

Para uma análise detalhada da origem desse exemplo, veja a publicação na [Comunidade da AWS](#).

Serviços utilizados neste exemplo

- API Gateway
- DynamoDB
- Lambda
- Amazon Rekognition
- Amazon S3
- Amazon SNS

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulte [Usando o Amazon SNS com um SDK AWS](#). Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

## Criar uma aplicação de exploração do Amazon Textract

Os exemplos de código a seguir mostram como explorar a saída do Amazon Textract por meio de uma aplicação interativa.

### JavaScript

#### SDK para JavaScript (v3)

Mostra como usar o AWS SDK for JavaScript para criar um aplicativo React que usa o Amazon Textract para extrair dados de uma imagem de documento e exibi-los em uma página da web interativa. Este exemplo é executado em um navegador da Web e requer uma identidade autenticada do Amazon Cognito como credenciais. Ele usa o Amazon Simple Storage Service (Amazon S3) para armazenamento e, para notificações, pesquisa uma fila do Amazon Simple Queue Service (Amazon SQS) que está inscrita em um tópico do Amazon Simple Notification Service (Amazon SNS).

Para obter o código-fonte completo e instruções sobre como configurar e executar, veja o exemplo completo em [GitHub](#).

Serviços utilizados neste exemplo

- Identidade do Amazon Cognito

- Amazon S3
- Amazon SNS
- Amazon SQS
- Amazon Textract

## Python

### SDK para Python (Boto3)

Mostra como usar o AWS SDK for Python (Boto3) com o Amazon Textract para detectar elementos de texto, formulário e tabela em uma imagem de documento. A imagem de entrada e a saída do Amazon Textract são mostradas em um aplicativo Tkinter que permite explorar os elementos detectados.

- Envie uma imagem de documento para o Amazon Textract e explore a saída dos elementos detectados.
- Envie imagens diretamente para o Amazon Textract ou por meio de um bucket do Amazon Simple Storage Service (Amazon S3).
- Use APIs assíncronas para iniciar um trabalho que publica uma notificação em um tópico do Amazon Simple Notification Service (Amazon SNS) quando o trabalho for concluído.
- Faça uma pesquisa em uma fila do Amazon Simple Queue Service (Amazon SQS) para obter uma mensagem de conclusão do trabalho e exiba os resultados.

Para obter o código-fonte completo e instruções sobre como configurar e executar, veja o exemplo completo em [GitHub](#).

### Serviços utilizados neste exemplo

- Amazon S3
- Amazon SNS
- Amazon SQS
- Amazon Textract

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulte [Usando o Amazon SNS com um SDK AWS](#). Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

## Detecte pessoas e objetos em um vídeo com o Amazon Rekognition usando um SDK AWS

Os exemplos de código a seguir mostram como detectar pessoas e objetos em um vídeo com o Amazon Rekognition.

### Python

#### SDK para Python (Boto3)

Use o Amazon Rekognition para detectar faces, objetos e pessoas em vídeos iniciando trabalhos de detecção assíncrona. Este exemplo também configura o Amazon Rekognition para notificar um tópico do Amazon Simple Notification Service (Amazon SNS) quando os trabalhos são concluídos e inscreve uma fila do Amazon Simple Queue Service (Amazon SQS) no tópico. Quando a fila recebe uma mensagem sobre um trabalho, o trabalho é recuperado e os resultados são apresentados.

Este exemplo é melhor visualizado em GitHub. Para obter o código-fonte completo e instruções sobre como configurar e executar, veja o exemplo completo em [GitHub](#).

#### Serviços utilizados neste exemplo

- Amazon Rekognition
- Amazon SNS
- Amazon SQS

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulte [Usando o Amazon SNS com um SDK AWS](#). Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

## Publique mensagens do Amazon SNS nas filas do Amazon SQS usando um SDK AWS

Os exemplos de código a seguir mostram como:

- Crie um tópico (FIFO ou não FIFO).
- Assinar várias filas no tópico com a opção de aplicar um filtro.
- Publicar mensagens no tópico.

- Pesquisar as filas para ver as mensagens recebidas.

## Java

### SDK para Java 2.x

Demonstra mensagens com tópicos e filas usando o Amazon Simple Notification Service (Amazon SNS) e o Amazon Simple Queue Service (Amazon SQS).

Para obter o código-fonte completo e as instruções que demonstram mensagens com tópicos e filas no Amazon SNS e no Amazon SQS, veja o exemplo completo em. [GitHub](#)

Serviços utilizados neste exemplo

- Amazon SNS
- Amazon SQS

## Kotlin

### SDK para Kotlin

Demonstra mensagens com tópicos e filas usando o Amazon Simple Notification Service (Amazon SNS) e o Amazon Simple Queue Service (Amazon SQS).

Para obter o código-fonte completo e as instruções que demonstram mensagens com tópicos e filas no Amazon SNS e no Amazon SQS, veja o exemplo completo em. [GitHub](#)

Serviços utilizados neste exemplo

- Amazon SNS
- Amazon SQS

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulte [Usando o Amazon SNS com um SDK AWS](#). Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

## Usar o API Gateway para invocar uma função do Lambda

Os exemplos de código a seguir mostram como criar uma AWS Lambda função invocada pelo Amazon API Gateway.

## Java

### SDK para Java 2.x

Mostra como criar uma AWS Lambda função usando a API de tempo de execução Lambda Java. Este exemplo invoca AWS serviços diferentes para realizar um caso de uso específico. Este exemplo mostra como criar uma função do Lambda invocada pelo Amazon API Gateway que verifica uma tabela do Amazon DynamoDB em busca de aniversários de trabalho e usa o Amazon Simple Notification Service (Amazon SNS) para enviar uma mensagem de texto aos seus funcionários que os parabeniza em sua data de aniversário de um ano.

Para obter o código-fonte completo e instruções sobre como configurar e executar, veja o exemplo completo em [GitHub](#).

Serviços utilizados neste exemplo

- API Gateway
- DynamoDB
- Lambda
- Amazon SNS

## JavaScript

### SDK para JavaScript (v3)

Mostra como criar uma AWS Lambda função usando a API de tempo de JavaScript execução do Lambda. Este exemplo invoca AWS serviços diferentes para realizar um caso de uso específico. Este exemplo mostra como criar uma função do Lambda invocada pelo Amazon API Gateway que verifica uma tabela do Amazon DynamoDB em busca de aniversários de trabalho e usa o Amazon Simple Notification Service (Amazon SNS) para enviar uma mensagem de texto aos seus funcionários que os parabeniza em sua data de aniversário de um ano.

Para obter o código-fonte completo e instruções sobre como configurar e executar, veja o exemplo completo em [GitHub](#).

Esse exemplo também está disponível no [Guia do desenvolvedor do AWS SDK for JavaScript v3](#).

## Serviços usados neste exemplo

- API Gateway
- DynamoDB
- Lambda
- Amazon SNS

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulte [Usando o Amazon SNS com um SDK AWS](#). Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

## Usar eventos programados para invocar uma função do Lambda

Os exemplos de código a seguir mostram como criar uma AWS Lambda função invocada por um evento EventBridge agendado pela Amazon.

### Java

#### SDK para Java 2.x

Mostra como criar um evento EventBridge programado pela Amazon que invoca uma AWS Lambda função. Configure EventBridge para usar uma expressão cron para agendar quando a função Lambda é invocada. Neste exemplo, você cria uma função do Lambda usando a API de runtime de Java do Lambda. Este exemplo invoca AWS serviços diferentes para realizar um caso de uso específico. Este exemplo mostra como criar uma aplicação que envia uma mensagem de texto móvel para seus funcionários que os parabeniza na data de aniversário de um ano.

Para obter o código-fonte completo e instruções sobre como configurar e executar, veja o exemplo completo em [GitHub](#).

## Serviços usados neste exemplo

- DynamoDB
- EventBridge
- Lambda
- Amazon SNS

## JavaScript

### SDK para JavaScript (v3)

Mostra como criar um evento EventBridge programado pela Amazon que invoca uma AWS Lambda função. Configure EventBridge para usar uma expressão cron para agendar quando a função Lambda é invocada. Neste exemplo, você cria uma função Lambda usando a API de tempo de execução do JavaScript Lambda. Este exemplo invoca AWS serviços diferentes para realizar um caso de uso específico. Este exemplo mostra como criar uma aplicação que envia uma mensagem de texto móvel para seus funcionários que os parabeniza na data de aniversário de um ano.

Para obter o código-fonte completo e instruções sobre como configurar e executar, veja o exemplo completo em [GitHub](#).

Esse exemplo também está disponível no [Guia do desenvolvedor do AWS SDK for JavaScript v3](#).

Serviços usados neste exemplo

- DynamoDB
- EventBridge
- Lambda
- Amazon SNS

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulte [Usando o Amazon SNS com um SDK AWS](#). Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

# Segurança do Amazon SNS

Esta seção fornece informações sobre a segurança, a autenticação e o controle de acesso do Amazon SNS, bem como a Linguagem de política de acesso do Amazon SNS.

## Tópicos

- [Proteção de dados](#)
- [Gerenciamento de identidade e acesso no Amazon SNS](#)
- [Registrar em log e monitorar no Amazon SNS](#)
- [Validação de conformidade para o Amazon SNS](#)
- [Resiliência no Amazon SNS](#)
- [Segurança da infraestrutura no Amazon SNS](#)
- [Práticas recomendadas de segurança para o Amazon SNS](#)

## Proteção de dados

O [modelo de responsabilidade compartilhada](#) da AWS se aplica à proteção de dados no Amazon Simple Notification Service. Conforme descrito nesse modelo, a AWS é responsável por proteger a infraestrutura global que executa toda a Nuvem AWS. Você é responsável por manter o controle sobre seu conteúdo hospedado nessa infraestrutura. Esse conteúdo inclui as tarefas de configuração e gerenciamento de segurança dos serviços da AWS que você usa. Para obter mais informações sobre a privacidade de dados, consulte as [Perguntas frequentes sobre privacidade de dados](#). Para obter mais informações sobre a proteção de dados na Europa, consulte a postagem do blog [AWS Shared Responsibility Model and GDPR](#) no Blog de segurança da AWS.

Para fins de proteção de dados, recomendamos que você proteja as credenciais da conta da Conta da AWS e configure as contas de usuário individuais com o AWS Identity and Access Management (IAM). Dessa maneira, cada usuário receberá apenas as permissões necessárias para cumprir suas obrigações de trabalho. Recomendamos também que você proteja seus dados das seguintes formas:

- Use uma autenticação multifator (MFA) com cada conta.
- Use SSL/TLS para se comunicar com os recursos da AWS. Recomendamos TLS 1.2 ou posterior.
- Configure o registro em log das atividades da API e do usuário com o AWS CloudTrail.

- Use as soluções de criptografia da AWS, juntamente com todos os controles de segurança padrão nos serviços da AWS.
- Use serviços gerenciados de segurança avançada, como o Amazon Macie, que ajuda a localizar e proteger dados pessoais armazenados no Amazon S3.
- Se você precisar de módulos criptográficos validados pelo FIPS 140-2 ao acessar a AWS por meio de uma interface de linha de comando ou uma API, use um endpoint do FIPS. Para obter mais informações sobre endpoints do FIPS, consulte [Federal Information Processing Standard \(FIPS\) 140-2](#).
- Proteção de dados de mensagens
  - A proteção de dados de mensagens é um recurso novo e importante do Amazon SNS.
  - Use o MDP para verificar mensagens quanto a informações sigilosas ou confidenciais.
  - Ofereça auditoria de mensagens para todo o conteúdo que flui pelo tópico.
  - Forneça controles de acesso ao conteúdo às mensagens publicadas no tópico e às mensagens entregues pelo tópico.

#### Important

É altamente recomendável que você nunca coloque informações de identificação confidenciais, como endereços de e-mail dos seus clientes, em marcações ou campos de formato livre, como um campo Name (Nome). Isso também vale para o uso do Amazon SNS ou de outros serviços da Amazon Web Services com o console, a API, a AWS CLI ou os AWS SDKs. Quaisquer dados inseridos em marcações ou campos de formato livre usados para nomes podem ser usados para logs de cobrança ou diagnóstico. Se você fornecer um URL para um servidor externo, recomendamos fortemente que não sejam incluídas informações de credenciais no URL para validar a solicitação a esse servidor.

As seções a seguir fornecem mais informações sobre a proteção de dados no Amazon SNS.

#### Tópicos

- [Criptografia de dados](#)
- [Privacidade do tráfego entre redes](#)
- [Segurança de proteção de dados de mensagens](#)

## Criptografia de dados

Proteção de dados protege os dados em trânsito (à medida que são transferidos para e do Amazon SNS) e em repouso (enquanto estão armazenados em discos em datacenters do Amazon SNS). Você pode proteger os dados em trânsito usando Secure Sockets Layer (SSL) ou criptografia no lado do cliente. Por padrão, o Amazon SNS armazena mensagens e arquivos usando criptografia de disco. Você pode proteger dados em repouso solicitando que o Amazon SNS criptografe suas mensagens antes de salvá-las no sistema de arquivos criptografados em seus datacenters. O Amazon SNS recomenda o uso do SSE para otimizar a criptografia de dados.

### Tópicos

- [Criptografia inativa](#)
- [Gerenciamento de chaves](#)
- [Habilitar a criptografia no lado do servidor \(SSE\) para um tópico do Amazon SNS](#)
- [Habilitar a criptografia no lado do servidor \(SSE\) para um tópico do Amazon SNS com uma fila do Amazon SQS criptografada inscrita](#)

### Criptografia inativa

A criptografia do lado do servidor (SSE) permite armazenar dados confidenciais em tópicos criptografados, protegendo o conteúdo das mensagens nos tópicos do Amazon SNS usando chaves gerenciadas em (). AWS Key Management Service AWS KMS

A SSE criptografa mensagens assim que o Amazon SNS as recebe. As mensagens são armazenadas de forma criptografada e somente descriptografadas quando são enviadas.

- Para obter informações sobre como gerenciar a SSE usando o AWS Management Console ou o AWS SDK for Java (ao configurar o atributo `KmsMasterKeyId` usando as ações de API [CreateTopic](#) e [SetTopicAttributes](#)), consulte [Habilitar a criptografia no lado do servidor \(SSE\) para um tópico do Amazon SNS](#).
- Para obter informações sobre como criar tópicos criptografados usando AWS CloudFormation (ao definir a propriedade `KmsMasterKeyId` usando o recurso [AWS::SNS::Topic](#)), consulte o Manual do usuário do AWS CloudFormation.

**⚠ Important**

Todas as solicitações para tópicos com SSE ativada devem usar HTTPS e o [Signature versão 4](#).

Para obter informações sobre a compatibilidade de outros serviços com filas criptografadas, consulte a documentação do seu serviço.

O Amazon SNS só é compatível com chaves do KMS de criptografia simétrica. Você não pode usar nenhum outro tipo de chave do KMS para criptografar seus recursos de serviço.

Para obter ajuda para determinar se uma chave do KMS é simétrica, consulte [Identificar chaves assimétricas do KMS](#).

O AWS KMS combina hardware e software seguros e altamente disponíveis para fornecer um sistema de gerenciamento de chaves escalado para a nuvem. Quando você usa o Amazon SNS com o AWS KMS, as [chaves de dados](#) que criptografam os dados da mensagem também são criptografadas e armazenadas com os dados que protegem.

Os benefícios de usar o AWS KMS são os seguintes:

- Você pode criar e gerenciar o [AWS KMS key](#) por conta própria.
- Você também pode usar chaves do KMS gerenciadas pela AWS para o Amazon SNS, que são exclusivas para cada conta e região.
- Os padrões de segurança do AWS KMS podem ajudá-lo a atender aos requisitos de compatibilidade relacionados à criptografia.

Para obter mais informações, consulte [O que é o AWS Key Management Service?](#) no Guia do desenvolvedor do AWS Key Management Service.

## Tópicos

- [Escopo de criptografia](#)
- [Principais termos](#)

## Escopo de criptografia

A SSE criptografa o corpo de uma mensagem em um tópico do Amazon SNS.

A SSE não criptografa o seguinte:

- Metadados de tópico (nome e atributos do tópico)
- Metadados de mensagens (assunto, ID de mensagem, carimbo de data/hora e atributos)
- Política de proteção de dados
- Métricas por tópico

#### Note

- Uma mensagem só será criptografada se for enviada após a habilitação da criptografia de um tópico. O Amazon SNS não criptografa mensagens com lista de pendências.
- Qualquer mensagem criptografada permanecerá dessa forma mesmo se a criptografia de seu tópico for desabilitada.

## Principais termos

Os seguintes termos-chave podem ajudar você a entender melhor a funcionalidade da SSE. Para obter descrições detalhadas, consulte a [Referência da API do Amazon Simple Notification Service](#).

### Chave de dados

A chave de criptografia dos dados (DEK) responsável por criptografar o conteúdo de mensagens do Amazon SNS.

Para obter mais informações, consulte [Data Keys](#) (“Chaves de dados”) no Guia do desenvolvedor do AWS Key Management Service e [Envelope Encryption](#) (“Criptografia de envelope”) no Guia do desenvolvedor do AWS Encryption SDK.

### ID do AWS KMS key

O alias, o ARN do alias, o ID de chave ou o ARN de uma AWS KMS key ou um AWS KMS personalizado: em sua conta ou em outra. Embora o alias do AWS KMS gerenciado pela AWS para o Amazon SNS seja sempre `alias/aws/sns`, o alias de um AWS KMS personalizado pode, por exemplo, ser `alias/MyAlias`. Você pode usar essas chaves do AWS KMS para proteger as mensagens em tópicos do Amazon SNS.

#### Note

Lembre-se do seguinte:

- Na primeira vez em que você usar o AWS Management Console para especificar o KMS gerenciado pela AWS para o Amazon SNS referente a um tópico, o AWS KMS criará o KMS gerenciado pela AWS para o Amazon SNS.
- Como alternativa, na primeira vez em que você usar a ação Publish em um tópico com SSE habilitada, o AWS KMS criará o KMS gerenciado pela AWS para o Amazon SNS.

Você pode criar chaves do AWS KMS, definir as políticas que controlam como as chaves do AWS KMS podem ser usadas e auditar o uso do AWS KMS utilizando a seção AWS KMS keys do console do AWS KMS ou a ação AWS KMS do [CreateKey](#). Para obter mais informações, consulte [AWS KMS keys](#) e [Criação de chaves](#) no Guia do desenvolvedor do AWS Key Management Service. Para ver mais exemplos de AWS KMS identificadores, consulte [KeyId](#) da Referência da AWS Key Management Service API. Para obter informações sobre como encontrar identificadores do AWS KMS, consulte [Encontrar o ID da chave e o ARN](#) no Guia do desenvolvedor do AWS Key Management Service.

#### Important

Não há custos adicionais por usar o AWS KMS. Para obter mais informações, consulte [Estimar os custos do AWS KMS](#) e [Definição de preço do AWS Key Management Service](#).

## Gerenciamento de chaves

As seções a seguir contêm informações sobre como trabalhar com chaves gerenciadas pelo AWS Key Management Service (AWS KMS). Para saber mais sobre

#### Note

O Amazon SNS só é compatível com chaves do KMS de criptografia simétrica. Você não pode usar nenhum outro tipo de chave do KMS para criptografar seus recursos de serviço. Para obter ajuda para determinar se uma chave do KMS é simétrica, consulte [Identificar chaves assimétricas do KMS](#).

## Tópicos

- [Estimar os custos do AWS KMS](#)
- [Configurar permissões do AWS KMS](#)
- [Erros do AWS KMS](#)

## Estimar os custos do AWS KMS

Para prever custos e entender melhor a sua fatura da AWS, talvez seja interessante saber com que frequência o Amazon SNS usa sua chave do AWS KMS key.

### Note

Embora a fórmula a seguir possa dar a você uma boa ideia sobre os custos esperados, os custos reais poderão ser mais altos por conta da natureza distribuída do Amazon SNS.

Para calcular o número de solicitações de APIs (R) por tópico, use a seguinte fórmula:

$$R = B / D * (2 * P)$$

B é o período de faturamento (em segundos).

D é o período de reutilização da chave de dados (em segundos: o Amazon SNS reutiliza uma chave de dados por até cinco minutos).

P é o número de [principais](#) de publicação enviados para o tópico do Amazon SNS.

Estes são cálculos de exemplo. Para obter informações sobre a definição de preços, consulte [Definição de preços do AWS Key Management Service](#).

Exemplo 1: Cálculo do número de chamadas à API do AWS KMS para 1 editor e 1 tópico

Este exemplo supõe o seguinte:

- O período de faturamento é de 1 a 31 de janeiro (2.678.400 segundos).
- O período de reutilização de chave de dados é de 5 minutos (300 segundos).
- Há 1 tópico.
- Há 1 entidade principal de publicação.

$$2,678,400 / 300 * (2 * 1) = 17,856$$

Exemplo 2: Cálculo do número de chamadas à API do AWS KMS para vários editores e 2 tópicos

Este exemplo supõe o seguinte:

- O período de faturamento é de 1 a 28 de fevereiro (2.419.200 segundos).
- O período de reutilização de chave de dados é de 5 minutos (300 segundos).
- Há 2 tópicos.
- O primeiro tópico tem 3 entidades principais de publicação.
- O segundo tópico tem 5 entidades principais de publicação.

$$(2,419,200 / 300 * (2 * 3)) + (2,419,200 / 300 * (2 * 5)) = 129,024$$

## Configurar permissões do AWS KMS

Antes que possa usar a SSE, você deve configurar as políticas de chaves do AWS KMS key para permitir a criptografia de tópicos, bem como a criptografia e a descryptografia de mensagens.

Para obter exemplos e mais informações sobre as permissões do AWS KMS, consulte [AWS KMS API Permissions: Actions and Resources Reference](#) no Guia do desenvolvedor do AWS Key Management Service. Para obter detalhes sobre como configurar um tópico do Amazon SNS com a criptografia do lado do servidor, consulte [Configurar um tópico do Amazon SNS com a criptografia do lado do servidor](#).

### Note

Você também pode gerenciar permissões para criptografia simétrica de chaves do KMS usando políticas do IAM. Para obter mais informações, consulte [Uso de políticas do IAM com o AWS KMS](#).

Embora seja possível configurar permissões globais para envio e recebimento do Amazon SNS, o AWS KMS exige a indicação explícita do ARN completo dos KMSs em regiões específicas na seção Resource de uma política do IAM.

É necessário garantir também que as políticas de chave do AWS KMS key concedam as permissões necessárias. Para fazer isso, indique as entidades principais que produzem e consomem mensagens criptografadas no Amazon SNS como usuários na política de chaves do KMS.

Como alternativa, você pode especificar as ações necessárias do AWS KMS e o ARN do KMS em uma política do IAM atribuída às entidades principais que publicam e se inscrevem para receber mensagens criptografadas no Amazon SNS. Para obter mais informações, consulte [Gerenciar o acesso ao AWS KMS](#) no Guia do desenvolvedor do AWS Key Management Service.

Se você estiver selecionando uma chave gerenciada pelo cliente para seu tópico do Amazon SNS e estiver usando aliases para controlar o acesso às chaves do KMS usando políticas do IAM ou políticas de chave do KMS com a chave de condição `kms:ResourceAliases`, a chave gerenciada pelo cliente selecionada também deverá ter um alias associado. Para obter mais informações sobre o uso de alias para controlar o acesso às chaves do KMS, consulte [Usar aliases para controlar o acesso às chaves do KMS](#) no Guia do desenvolvedor do AWS Key Management Service.

Permitir que um usuário envie mensagens a um tópico com SSE

O editor deve ter as permissões `kms:GenerateDataKey*` e `kms:Decrypt` para o AWS KMS key.

```
{
  "Statement": [{
    "Effect": "Allow",
    "Action": [
      "kms:GenerateDataKey*",
      "kms:Decrypt"
    ],
    "Resource": "arn:aws:kms:us-
east-2:123456789012:key/1234abcd-12ab-34cd-56ef-1234567890ab"
  }, {
    "Effect": "Allow",
    "Action": [
      "sns:Publish"
    ],
    "Resource": "arn:aws:sns:*:123456789012:MyTopic"
  }]
}
```

## Habilitar a compatibilidade entre as fontes de eventos de tópicos criptografados e de serviços da AWS

Vários serviços da AWS publicam eventos nos tópicos do Amazon SNS. Para permitir que essas fontes de eventos trabalhem com tópicos criptografados, você deve executar as seguintes etapas:

1. Use uma chave gerenciada pelo cliente. Para obter mais informações, consulte [Criação de chaves](#) no Guia do desenvolvedor AWS Key Management Service.

2. Para permitir que o serviço da AWS tenha as permissões `kms:GenerateDataKey*` e `kms:Decrypt`, adicione a seguinte instrução à política do KMS.

```
{
  "Statement": [{
    "Effect": "Allow",
    "Principal": {
      "Service": "service.amazonaws.com"
    },
    "Action": [
      "kms:GenerateDataKey*",
      "kms:Decrypt"
    ],
    "Resource": "*"
  }]
}
```

Origem do evento.	Principal do serviço
<a href="#">Amazon CloudWatch</a>	cloudwatch.amazonaws.com
<a href="#">Amazon CloudWatch Events</a>	events.amazonaws.com
<a href="#">AWS CodeCommit</a>	codecommit.amazonaws.com
<a href="#">AWS CodeStar</a>	codestar-notifications.amazonaws.com
<a href="#">AWS Database Migration Service</a>	dms.amazonaws.com
<a href="#">AWS Directory Service</a>	ds.amazonaws.com
<a href="#">Amazon DynamoDB</a>	dynamodb.amazonaws.com
<a href="#">Amazon Inspector</a>	inspector.amazonaws.com
<a href="#">Amazon Redshift</a>	redshift.amazonaws.com
<a href="#">Amazon RDS</a>	events.rds.amazonaws.com
<a href="#">Amazon S3 Glacier</a>	glacier.amazonaws.com

Origem do evento.	Principal do serviço
<a href="#">Amazon Simple Email Service</a>	ses.amazonaws.com
<a href="#">Amazon Simple Storage Service</a>	s3.amazonaws.com
<a href="#">AWS Snowball</a>	importexport.amazonaws.com
<a href="#">AWS Systems Manager Incident Manager</a>	O AWS Systems Manager Incident Manager consiste em dois princípios de serviço: ssm-incidents.amazonaws.com ; ssm-contacts.amazonaws.com

**Note**

Algumas fontes de eventos do Amazon SNS exigem que você forneça um perfil do IAM (em vez da entidade principal do serviço) na política do AWS KMS key:

- [Amazon EC2 Auto Scaling](#)
- [Amazon Elastic Transcoder](#)
- [AWS CodePipeline](#)
- [AWS Config](#)
- [AWS Elastic Beanstalk](#)
- [AWS IoT](#)
- [EC2 Image Builder](#)

3. Adicione as chaves de condição `aws:SourceAccount` e `aws:SourceArn` à política de recursos do KMS para proteger ainda mais a chave do KMS de ataques [confused deputy](#). Consulte a lista de documentação específica do serviço (acima) para obter detalhes exatos em cada caso.

**Important**

A adição de `aws:SourceAccount` e `aws:SourceArn` a uma política do AWS KMS não é compatível com tópicos criptografados do EventBridgeto.

```
{
  "Effect": "Allow",
  "Principal": {
    "Service": "service.amazonaws.com"
  },
  "Action": [
    "kms:GenerateDataKey*",
    "kms:Decrypt"
  ],
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "aws:SourceAccount": "customer-account-id"
    },
    "ArnLike": {
      "aws:SourceArn": "arn:aws:service:region:customer-account-id:resource-
type:customer-resource-id"
    }
  }
}
```

4. [Habilite a SSE para seu tópico](#) usando seu KMS.
5. Forneça o ARN do tópico criptografado para a fonte do evento.

## Erros do AWS KMS

Ao trabalhar com o Amazon SNS e o AWS KMS, você poderá encontrar erros. A lista a seguir descreve os erros e as possíveis soluções de problemas.

### KMSAccessDeniedException

O texto cifrado faz referência a uma chave que não existe ou à qual você não tem acesso.

Código de status HTTP: 400

### KMSDisabledException

A solicitação foi recusada porque o KMS especificado não está habilitado.

Código de status HTTP: 400

## KMSInvalidStateException

A solicitação foi rejeitada porque o estado do recurso especificado não é válido para essa solicitação. Para obter mais informações, consulte [Principais estados do AWS KMS keys](#) no Guia do desenvolvedor do AWS Key Management Service.

Código de status HTTP: 400

## KMSNotFoundException

A solicitação foi rejeitada porque a entidade ou o recurso especificado não pôde ser encontrado.

Código de status HTTP: 400

## KMSOptInRequired

O ID da chave de acesso da AWS precisa de uma assinatura do serviço.

Código de status HTTP: 403

## KMSThrottlingException

A solicitação foi negada devido à limitação da solicitação. Para obter mais informações sobre controle de utilização, consulte [Cotas](#) no Guia do desenvolvedor do AWS Key Management Service.

Código de status HTTP: 400

## Habilitar a criptografia no lado do servidor (SSE) para um tópico do Amazon SNS

Com a criptografia do lado do servidor (SSE), é possível armazenar dados confidenciais em tópicos criptografados. A SSE protege o conteúdo de mensagens nos tópicos do Amazon SNS usando chaves gerenciadas no AWS Key Management Service (AWS KMS). Para obter mais informações sobre a criptografia do lado do servidor com o Amazon SNS, consulte [Criptografia inativa](#). Para saber mais sobre como criar chaves do AWS KMS, consulte [Criar chaves](#) no Guia do desenvolvedor do AWS Key Management Service.

### Important

Todas as solicitações para tópicos com SSE ativada devem usar HTTPS e o [Signature versão 4](#).

## Habilitar a criptografia do lado do servidor (SSE) para um tópico do Amazon SNS usando o AWS Management Console

1. Faça login no [console do Amazon SNS](#).
2. No painel de navegação, escolha Topics (Tópicos).
3. Na página Topics (Tópicos), selecione um tópico e escolha Actions (Ações), Edit (Editar).
4. Expanda a seção Encryption (Criptografia) e faça o seguinte:
  - a. Selecione Enable encryption (Ativar criptografia).
  - b. Especifique a chave do AWS KMS. Para obter mais informações, consulte [Principais termos](#).

Para cada tipo de KMS, são exibidos Description (Descrição), Account (Conta) e KMS ARN (ARN do KMS).

### Important

Se você não for o proprietário do KMS ou se fizer login com uma conta que não tenha as permissões `kms:ListAliases` e `kms:DescribeKey`, não será possível visualizar as informações sobre o KMS no console do Amazon SNS.

Peça ao proprietário do KMS para conceder essas permissões a você. Para obter mais informações, consulte [AWS KMS API Permissions: Actions and Resources Reference](#) (“Permissões da API do KMS: referência de ações e recursos”) no Guia do desenvolvedor do AWS Key Management Service.

- O KMS gerenciado pela AWS para o Amazon SNS, (Default) `alias/aws/sns` [(Padrão) `alias/aws/sns`] é selecionado por padrão.

### Note

Lembre-se do seguinte:

- Na primeira vez em que você usar o AWS Management Console para especificar o KMS gerenciado pela AWS para o Amazon SNS referente a um tópico, o AWS KMS criará o KMS gerenciado pela AWS para o Amazon SNS.

- Como alternativa, na primeira vez em que você usar a ação Publish em um tópico com SSE habilitada, o AWS KMS criará o KMS gerenciado pela AWS para o Amazon SNS.

- Para usar uma chave personalizada do KMS de sua conta da AWS, escolha o campo Chave do KMS e selecione a chave personalizada do KMS na lista.

 Note

Para obter instruções sobre como criar KMSs personalizados, consulte [Criar chaves](#) no Guia do desenvolvedor do AWS Key Management Service.

- Para usar um ARN personalizado do KMS de sua conta da AWS ou de outra conta da AWS, insira-o no campo Chave do KMS.

5. Escolha Save changes (Salvar alterações).

SSE é habilitada para o seu tópico e a página **MeuTópico** é exibida.

O status de Criptografia, a Conta da AWS, a Chave mestra do cliente (CMK), o ARN da CMK e a Descrição do tópico são exibidos na guia Criptografia.

Configurar um tópico do Amazon SNS com a criptografia do lado do servidor

Ao criar a chave do KMS, use a seguinte política de chave do KMS:

```
{
  "Effect": "Allow",
  "Principal": {
    "Service": "service.amazonaws.com"
  },
  "Action": [
    "kms:Decrypt",
    "kms:GenerateDataKey"
  ],
  "Resource": "*",
  "Condition": {
    "ArnLike": {
      "aws:SourceArn": "arn:aws:service:region:customer-account-id:resource-type/customer-resource-id"
    }
  },
}
```

```
    "StringEquals": {
      "kms:EncryptionContext:aws:sns:topicArn":
"arn:aws:sns:your_region:customer-account-id:your_sns_topic_name"
    }
  }
}
```

## Habilitar a criptografia no lado do servidor (SSE) para um tópico do Amazon SNS com uma fila do Amazon SQS criptografada inscrita

Você pode habilitar a criptografia no lado do servidor (SSE) para um tópico com o intuito de proteger seus dados. Para permitir que o Amazon SNS envie mensagens para filas do Amazon SQS criptografadas, a chave mestra de cliente (CMK) associada à fila do Amazon SQS deve ter uma declaração de política que conceda ao Amazon SNS o acesso principal ao serviço para as ações da API do AWS KMS, `GenerateDataKey` e `Decrypt`. Para obter mais informações sobre como usar a SSE, consulte [Criptografia inativa](#).

Esta página mostra como é possível habilitar o SSE para um tópico do Amazon SNS no qual uma fila criptografada do Amazon SQS está inscrita usando o AWS Management Console.

### Etapa 1: como criar uma chave do KMS

1. Faça login no [console do AWS KMS](#) com um usuário que tenha pelo menos a política `AWSKeyManagementServicePowerUser`.
2. Escolha `Create a key` (Criar uma chave).
3. Para criar uma chave do KMS de criptografia simétrica, em `Key type` (Tipo de chave), selecione `Symmetric` (Simétrica).

Para obter informações sobre como criar uma chave do KMS assimétrica no console do AWS KMS, consulte [Criar chaves do KMS assimétricas \(console\)](#).

4. Em `Key usage` (Uso da chave), a opção `Encrypt and decrypt` (Criptografar e descriptografar) é selecionada para você.

Para obter informações sobre como criar chaves do KMS que geram e verificam códigos MAC, consulte [Criar chaves do KMS de HMAC](#).

Para obter mais informações sobre as opções avançadas, consulte [Chaves para fins especiais](#).

5. Escolha `Next` (Próximo).

6. Digite um alias para a chave do KMS. O nome do alias não pode começar com **aws/**. O prefixo **aws/** é reservado pela Amazon Web Services para representar as Chaves gerenciadas pela AWS na sua conta.

 Note

Adicionar, excluir ou atualizar um alias pode conceder ou negar uma permissão à chave do KMS. Para obter detalhes, consulte [ABAC para AWS KMS](#) e [Uso de aliases para controlar o acesso às chaves do KMS](#).

Um alias é um nome de exibição que identifica a chave do KMS. Recomendamos que você escolha um alias que indique o tipo de dados que pretende proteger ou a aplicação a ser usada com a chave do KMS.

Aliases são necessários ao criar uma chave do KMS no AWS Management Console. Eles são opcionais quando você usa a operação [CreateKey](#).

7. (Opcional) Digite uma descrição para a chave do KMS.

Você pode adicionar uma descrição agora ou atualizá-la a qualquer momento, a não ser que o [estado da chave](#) seja Pending Deletion ou Pending Replica Deletion. Para adicionar, alterar ou excluir a descrição de uma chave gerenciada pelo cliente existente, [edite a descrição](#) na operação AWS Management Console ou use a operação [UpdateKeyDescription](#).

8. (Opcional) Digite uma chave de tag e um valor de tag opcional. Para adicionar mais de uma etiqueta à chave do KMS, selecione Add tag (Adicionar etiqueta).

 Note

Marcar ou desmarcar uma chave do KMS pode conceder ou negar uma permissão a essa chave do KMS. Para obter detalhes, consulte [ABAC para AWS KMS](#) e [Uso de tags para controlar o acesso às chaves do KMS](#).

Ao adicionar tags aos recursos da AWS, a AWS gera um relatório de alocação de custos com utilização e custos agrupados por tags. Etiquetas também podem ser utilizadas para controlar o acesso a uma chave do KMS. Para informações sobre marcação de chaves do KMS, consulte [Marcação de chaves](#) e [ABAC para AWS KMS](#).

9. Escolha Next (Próximo).
10. Selecione os usuários e as funções do IAM que podem administrar a chave do KMS.

 Note

Esta política de chave concede controle total dessa chave do KMS à Conta da AWS. Ela permite que os administradores de conta usem políticas do IAM para conceder a outras entidades principais a permissão para gerenciar a chave do KMS. Para obter mais detalhes, consulte [Política de chaves padrão](#).

As práticas recomendadas do IAM não encorajam o uso de usuários do IAM com credenciais de longo prazo. Sempre que possível, use os perfis do IAM, por fornecerem credenciais temporárias. Para obter detalhes, consulte [Práticas recomendadas de segurança no IAM](#) no Guia do usuário do IAM.

11. (Opcional) Para evitar que os usuários e funções do IAM selecionados excluam essa chave do KMS, na seção Key deletion (Exclusão de chaves) na parte inferior da página, desmarque a caixa de seleção Allow key administrators to delete this key (Permitir que os administradores de chaves excluam essa chave).
12. Escolha Next (Próximo).
13. Selecione os usuários e os perfis do IAM que podem usar a chave em [operações de criptografia](#). Escolha Next (Próximo).
14. Na página Review and edit key policy (Revisar e editar política de chaves), adicione a seguinte instrução à política de chaves e, em seguida, escolha Finish (Concluir).

```
{
  "Sid": "Allow Amazon SNS to use this key",
  "Effect": "Allow",
  "Principal": {
    "Service": "sns.amazonaws.com"
  },
  "Action": [
    "kms:Decrypt",
    "kms:GenerateDataKey*"
  ],
  "Resource": "*"
}
```

Sua nova chave gerenciada pelo cliente será exibida na lista de chaves.

Etapa 2: como criar um tópico do Amazon SNS criptografado

1. Faça login no [console do Amazon SNS](#).
2. No painel de navegação, escolha Topics (Tópicos).
3. Escolha Create topic.
4. Na página Create new topic (Criar novo tópico), em Name (Nome), digite um nome para o tópico (por exemplo, MyEncryptedTopic) e escolha Create topic (Criar tópico).
5. Expanda a seção Encryption (Criptografia) e faça o seguinte:
  - a. Escolha Enable server-side encryption (Habilitar criptografia no lado do servidor).
  - b. Especifique a chave gerenciada pelo cliente. Para obter mais informações, consulte [Principais termos](#).

Para cada tipo de chave gerenciada pelo cliente, a Descrição, a Conta e o ARN da chave gerenciada pelo cliente são exibidos.

 Important

Se você não for o proprietário da chave gerenciada pelo cliente ou se fizer login com uma conta que não tenha as permissões `kms:ListAliases` e `kms:DescribeKey`, não será possível visualizar as informações sobre a chave gerenciada pelo cliente no console do Amazon SNS.

Peça ao proprietário da chave gerenciada pelo cliente para conceder essas permissões a você. Para obter mais informações, consulte [AWS KMS API Permissions: Actions and Resources Reference](#) (“Permissões da API do KMS: referência de ações e recursos”) no Guia do desenvolvedor do AWS Key Management Service.

- c. Para chave gerenciada pelo cliente, escolha MyCustomKey [que você criou anteriormente](#) e selecione Habilitar criptografia no lado do servidor.
6. Escolha Save changes (Salvar alterações).

SSE é habilitada para o seu tópico e a página MeuTópico é exibida.

O status de Criptografia, a Conta da AWS, a Chave mestra do cliente, o ARN e a Descrição do tópico são exibidos na guia Criptografia.

Seu novo tópico criptografado será exibido na lista de tópicos.

Etapa 3: como criar e se inscrever em filas criptografadas do Amazon SQS

1. Faça login no [console do Amazon SQS](#).
2. Selecione Create New Queue (Criar nova fila).
3. Na página Create New Queue (Criar nova fila), faça o seguinte:
  - a. Insira um Queue Name (Nome da fila) (por exemplo, MyEncryptedQueue1).
  - b. Escolha Standard Queue (Fila padrão) e, em seguida, escolha Configure Queue (Configurar fila).
  - c. Selecione Use SSE (Usar SSE).
  - d. Em AWS KMS key, escolha MyCustomKey [que você criou anteriormente](#), depois selecione Criar fila.
4. Repita o processo para criar uma segunda fila (por exemplo, chamada MyEncryptedQueue2).

Suas novas filas criptografadas serão exibidas na lista de filas.

5. No console do Amazon SQS, selecione MyEncryptedQueue1 e MyEncryptedQueue2, e escolha Queue Actions (Ações da fila), Subscribe Queues to SNS Topic (Inscrever filas em um tópico do SNS).
6. Na caixa de diálogo Subscribe to a Topic (Inscrever-se em um tópico), em Choose a topic (Escolher um tópico), selecione MyEncryptedTopic e, em seguida, escolha Subscribe (Inscrever-se).

Suas inscrições das filas criptografadas para o seu tópico criptografado são exibidas na caixa de diálogo Topic Subscription Result (Resultado de inscrição do tópico).

7. Escolha OK.

Etapa 4: como publicar uma mensagem no seu tópico criptografado

1. Faça login no [console do Amazon SNS](#).
2. No painel de navegação, escolha Topics (Tópicos).
3. Na lista de tópicos, selecione MyEncryptedTopic (Meu tópico criptografado) e Publish message (Publicar mensagem).
4. Na página Publish a message (Publicar uma mensagem), faça o seguinte:

- a. (Opcional) Na seção Message details (Detalhes da mensagem), insira o Subject (Assunto) (por exemplo, `Testing message publishing`).
- b. Na seção Message body (Corpo da mensagem), insira o corpo da mensagem (por exemplo, `My message body is encrypted at rest.`).
- c. Selecione Publish message (Publicar mensagem).

Sua mensagem será publicada nas suas filas criptografadas inscritas.

#### Etapa 5: como verificar entrega de mensagens

1. Faça login no [console do Amazon SQS](#).
2. Na lista de filas, escolha MyEncryptedQueue1 e selecione Send and receive messages (Enviar e receber mensagens).
3. Na página Send and receive messages in MyEncryptedQueue1 (Enviar e receber mensagens em MyEncryptedQueue1), escolha Poll for messages (Pesquisar mensagens).

A mensagem [que você enviou anteriormente](#) será exibida.

4. Escolha More Details (Mais detalhes) para visualizar sua mensagem.
5. Quando terminar, escolha Close (Fechar).
6. Repita o processo para MyEncryptedQueue2.

## Privacidade do tráfego entre redes

Um endpoint da Virtual Private Cloud (Amazon VPC) para Amazon SNS é uma entidade lógica dentro de uma VPC que permite conectividade apenas com o Amazon SNS. A VPC roteia as solicitações para o Amazon SNS e as respostas de volta para a VPC. As seções a seguir contêm informações sobre como trabalhar com VPC endpoints e criar políticas de VPC endpoint.

Se você usar a Amazon Virtual Private Cloud (Amazon VPC) para hospedar os recursos da AWS, poderá estabelecer uma conexão privada entre o VPC e o Amazon SNS. Com essa conexão, você pode publicar mensagens nos tópicos do Amazon SNS sem enviá-las pela Internet pública.

A Amazon VPC é um serviço da AWS que você pode usar para executar os recursos da AWS em uma rede virtual definida por você. Com a VPC, você tem controle sobre as configurações de rede, como o intervalo de endereços IP, sub-redes, tabelas de rotas e gateways de rede. Para conectar a VPC ao Amazon SNS, você define um VPC endpoint de interface. Esse tipo de endpoint permite

que você conecte a VPC aos serviços da AWS. O endpoint fornece uma conectividade confiável e escalável ao Amazon SNS sem a necessidade de um gateway da Internet, de uma instância de conversão de endereço de rede (NAT) ou de uma conexão VPN. Para obter mais informações, consulte [Interface VPC Endpoints](#) (Endpoints da VPC da interface) no Guia do usuário da Amazon VPC.

As informações nesta seção são para usuários da Amazon VPC. Para obter mais informações e para começar a criar uma VPC, consulte [Getting Started With Amazon VPC](#) (“Conceitos básicos do Amazon VPC”) no Manual do usuário da Amazon VPC.

#### Note

VPC endpoints não permitem que você se inscreva em um tópico do Amazon SNS para um endereço IP privado.

## Tópicos

- [Criar um endpoint da Amazon VPC para o Amazon SNS](#)
- [Criar uma política de endpoint da Amazon VPC para o Amazon SNS](#)
- [Publicar uma mensagem do Amazon SNS da Amazon VPC](#)

## Criar um endpoint da Amazon VPC para o Amazon SNS

Para publicar mensagens nos tópicos do Amazon SNS a partir de uma Amazon VPC, crie um VPC endpoint de interface. Em seguida, você poderá publicar mensagens nos seus tópicos, mantendo o tráfego na rede gerenciada com a VPC.

Use as informações a seguir para criar o endpoint e testar a conexão entre a VPC e o Amazon SNS. Consulte [Publicar uma mensagem do Amazon SNS da Amazon VPC](#) para ver instruções detalhadas sobre como começar do zero.

### Criar o endpoint

Você pode criar um endpoint do Amazon SNS em sua VPC usando o AWS Management Console, o, um AWS SDK AWS CLI, a API do Amazon SNS ou. AWS CloudFormation

Para obter informações sobre como criar e configurar um endpoint usando o console da Amazon VPC ou a AWS CLI, consulte [Creating an Interface Endpoint](#) (“Criar um endpoint da interface”) no Manual do usuário da Amazon VPC.

**⚠ Important**

Você pode usar a Amazon Virtual Private Cloud somente com endpoints HTTPS do Amazon SNS.

Ao criar um endpoint, especifique o Amazon SNS como o serviço ao qual a VPC deve se conectar. No console da Amazon VPC, os nomes de serviços variam de acordo com a região. Por exemplo, se você escolher Leste dos EUA (Norte da Virgínia), o nome do serviço será `com.amazonaws.us-east-1.sns`.

Ao configurar o Amazon SNS para enviar mensagens pela Amazon VPC, habilite o DNS privado e especifique endpoints no formato `sns.us-east-2.amazonaws.com`.

O DNS privado não oferece suporte a endpoints legados, como `queue.amazonaws.com` ou `us-east-2.queue.amazonaws.com`.

Para obter informações sobre como criar e configurar um endpoint usando AWS CloudFormation, consulte o [AWS::EC2::VPCEndpoint](#) recurso no Guia do AWS CloudFormation usuário.

Como testar a conexão entre a VPC e o Amazon SNS

Depois de criar um endpoint para o Amazon SNS, você pode publicar mensagens a partir da VPC nos tópicos do Amazon SNS. Para testar essa conexão, faça o seguinte:

1. Conecte-se a uma instância do Amazon EC2 que reside na VPC. Para obter mais informações sobre a conexão, consulte [Conecte-se à sua instância do Linux](#) ou [Conexão com sua instância Windows](#) na documentação do Amazon EC2.

Por exemplo, para se conectar a uma instância do Linux usando um cliente SSH, execute o comando a seguir a partir de um terminal:

```
$ ssh -i ec2-key-pair.pem ec2-user@instance-hostname
```

Em que:

- `ec2-key-pair.pem` é o arquivo que contém o par de chaves fornecido pelo Amazon EC2 quando você criou a instância.
- `instance-hostname` é o nome de host público da instância. Para obter o nome do host no [console do Amazon EC2](#): escolha Instances (Instâncias), selecione a instância e localize o valor de Public DNS (IPv4) (DNS público (IPv4)).

2. Na sua instância, use o comando [publish](#) do Amazon SNS com a AWS CLI. Você pode enviar uma mensagem simples para um tópico com o seguinte comando:

```
$ aws sns publish --region aws-region --topic-arn sns-topic-arn --message "Hello"
```

Em que:

- *aws-region* é a região em que AWS o tópico está localizado.
- *sns-topic-arn* é o Amazon Resource Name (ARN) do tópico. Para obter o ARN no [Console do Amazon SNS](#): escolha Topics (Tópicos), localize o tópico e encontre o valor na coluna ARN.

Se a mensagem for recebida com êxito pelo Amazon SNS, o terminal imprimirá um ID de mensagem, como o seguinte:

```
{
  "MessageId": "6c96dfff-0fdf-5b37-88d7-8cba910a8b64"
}
```

## Criar uma política de endpoint da Amazon VPC para o Amazon SNS

É possível criar uma política para endpoints da Amazon VPC para o Amazon SNS na qual se especifica o seguinte:

- O principal que pode executar ações.
- As ações que podem ser executadas.
- Os recursos sobre os quais as ações podem ser realizadas.

Para obter mais informações, consulte [Controlar o acesso a serviços com VPC endpoints](#) no Manual do usuário da Amazon VPC.

O exemplo de política de VPC endpoint a seguir especifica que o usuário MyUser do IAM tem permissão para publicar no tópico MyTopic do Amazon SNS.

```
{
  "Statement": [{
    "Action": ["sns:Publish"],
    "Effect": "Allow",
```

```
"Resource": "arn:aws:sns:us-east-2:123456789012:MyTopic",
"Principal": {
  "AWS": "arn:aws:iam:123456789012:user/MyUser"
}
}]
}
```

O seguinte é negado:

- Outras ações de API do Amazon SNS, como `sns:Subscribe` e `sns:Unsubscribe`.
- Outros usuários e regras do IAM que tentam usar esse endpoint da VPC.
- `MyUser` publicando em um tópico diferente do Amazon SNS.

#### Note

O usuário do IAM ainda pode usar outras ações de API do Amazon SNS de fora da VPC.

## Publicar uma mensagem do Amazon SNS da Amazon VPC

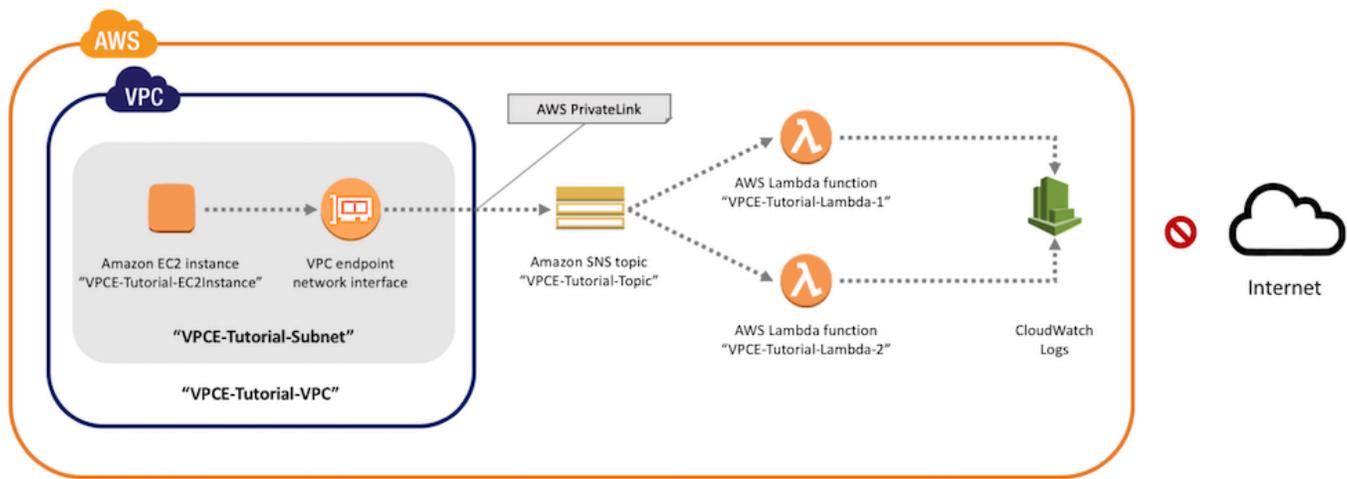
Esta seção descreve como publicar em um tópico do Amazon SNS, mantendo as mensagens seguras em uma rede privada. Publique uma mensagem a partir de uma instância do Amazon EC2 hospedada na Amazon Virtual Private Cloud (Amazon VPC). A mensagem permanecerá na rede da AWS sem ir para a Internet pública. Ao publicar mensagens de forma privada a partir de uma VPC, você pode melhorar a segurança do tráfego entre seus aplicativos e o Amazon SNS. Essa segurança é importante ao publicar informações de identificação pessoal (PII) sobre seus clientes ou quando sua assinatura está sujeita a regulamentações de mercado. Por exemplo, publicar de maneira privada é útil se você tiver um sistema para área da saúde que precise estar em conformidade com a Lei de Portabilidade e Responsabilidade de Provedores de Saúde (HIPAA) ou um sistema financeiro que precise estar em conformidade com o Padrão de segurança de dados do setor de cartão de pagamento (Payment Card Industry Data Security Standard, PCI DSS).

As etapas gerais são as seguintes:

- Use um modelo do AWS CloudFormation para criar automaticamente uma rede privada temporária em sua Conta da AWS.
- Crie um VPC endpoint que conecte a VPC com o Amazon SNS.

- Faça login em uma instância do Amazon EC2 e publique uma mensagem de maneira privada em um tópico do Amazon SNS.
- Verifique se a mensagem foi entregue com sucesso.
- Exclua os recursos que você criou durante este processo para que eles não permaneçam na sua Conta da AWS.

O diagrama seguinte descreve a rede privada que você cria na sua conta da AWS ao concluir essas etapas:



Esta rede consiste em uma VPC que contém uma instância do Amazon EC2. A instância se conecta ao Amazon SNS por meio de um endpoint da VPC de interface. Esse tipo de endpoint se conecta a serviços com tecnologia AWS PrivateLink. Com essa conexão estabelecida, você pode fazer login na instância do Amazon EC2 e publicar mensagens no tópico do Amazon SNS, mesmo que a rede esteja desconectada da Internet pública. O tópico envia as mensagens recebidas para duas funções AWS Lambda de assinatura. Essas funções registram as mensagens recebidas no Amazon CloudWatch Logs.

Demora cerca de 20 minutos para concluir essas etapas.

## Tópicos

- [Antes de começar](#)
- [Etapa 1: criar um par de chaves do Amazon EC2](#)
- [Etapa 2: Criar os recursos do AWS](#)
- [Etapa 3: verificar se a instância do Amazon EC2 não tem acesso à Internet](#)

- [Etapa 4: criar um endpoint da VPC para Amazon SNS](#)
- [Etapa 5: publicar uma mensagem no seu tópico do Amazon SNS](#)
- [Etapa 6: Verificar as entregas de mensagens](#)
- [Etapa 7: limpeza](#)
- [Recursos relacionados](#)

## Antes de começar

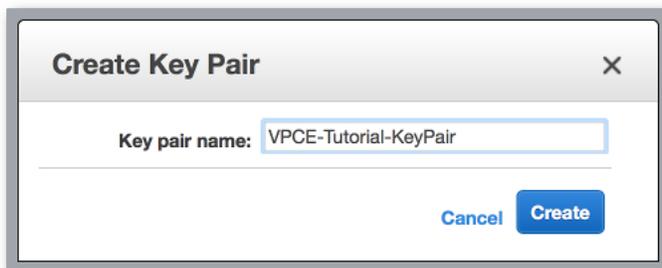
Antes de começar, você precisa de uma conta da Amazon Web Services (AWS). Ao cadastrar-se, sua conta é automaticamente cadastrada em todos os serviços da AWS incluindo o Amazon SNS e a Amazon VPC. Caso ainda não tenha criado uma conta, acesse <https://aws.amazon.com/> e escolha Create a Free Account (Criar conta gratuita).

## Etapa 1: criar um par de chaves do Amazon EC2

Um par de chaves é usado para fazer login em um instância do Amazon EC2. Esse par consiste em uma chave pública usada para criptografar suas informações de login e uma chave privada usada para descriptografá-la. Ao criar um par de chaves, você faz download de uma cópia da chave privada. Posteriormente, você usará o par de chaves para fazer login em um instância do Amazon EC2. Para efetuar login, especifique o nome do par de chaves e insira a chave privada.

## Para criar o par de chaves

1. Faça login no AWS Management Console e abra o console do Amazon EC2 em <https://console.aws.amazon.com/ec2/>.
2. No menu de navegação à esquerda, localize a seção Network & Security (Rede e segurança). Em seguida, escolha Key Pairs (Pares de chave).
3. Escolha Criar par de chaves.
4. Na janela Create Key Pair (Criar par de chaves), em Key pair name (Nome do par de chaves), digite **VPCE-Tutorial-KeyPair**. Selecione Create (Criar).



5. O arquivo de chave privada é baixado automaticamente pelo navegador. Salve-o em um local seguro. O Amazon EC2 dá a extensão ao arquivo .pem.
6. (Opcional) Se estiver usando um cliente de SSH em um computador Mac ou Linux para se conectar à sua instância, use o seguinte comando `chmod` para definir as permissões do arquivo de chave privada, de maneira que apenas você possa lê-lo:
  - a. Abra um terminal e navegue até o diretório que contém a chave privada:

```
$ cd /filepath_to_private_key/
```

- b. Defina as permissões usando o seguinte comando:

```
$ chmod 400 VPCE-Tutorial-KeyPair.pem
```

## Etapa 2: Criar os recursos do AWS

Para configurar a infraestrutura que oferece suporte a este tutorial, use um modelo do AWS CloudFormation. Um modelo é um arquivo que funciona como um plano para a criação de recursos da AWS, como instâncias do Amazon EC2 e tópicos do Amazon SNS. O modelo deste processo está disponível para download no GitHub.

Você fornece o modelo ao AWS CloudFormation, e o AWS CloudFormation provisiona os recursos necessários como uma pilha em sua Conta da AWS. Uma pilha é um conjunto de recursos que pode gerenciar como uma unidade. Ao concluir estas etapas, você poderá usar o AWS CloudFormation para excluir todos os recursos da pilha de uma só vez. Esses recursos não permanecerão na sua Conta da AWS, a menos que você queira.

A pilha deste processo inclui os seguintes recursos:

- Uma VPC e os recursos de rede associados, incluindo uma sub-rede, um grupo de segurança, um gateway da Internet e uma tabela de rotas.
- Uma instância do Amazon EC2 que é lançada na sub-rede na VPC.
- Um tópico do Amazon SNS.
- Duas funções AWS Lambda. Estas funções recebem mensagens que são publicadas no tópico do Amazon SNS e registram os eventos em log no CloudWatch Logs.
- Métricas e logs do Amazon CloudWatch.

- Uma função do IAM que permite que a instância do Amazon EC2 use o Amazon SNS e uma função do IAM que permite que as funções do Lambda gravem nos logs do CloudWatch.

Para criar recursos da AWS

1. Faça download do [arquivo do modelo](#) no site do GitHub.
2. Faça login no [AWS CloudFormation console](#).
3. Selecione Criar Stack.
4. Na página Select Template (Selecionar modelo), escolha Upload a template to Amazon S3 (Fazer upload de um modelo no Amazon S3), selecione o arquivo e Next (Avançar).
5. Na página Specify Details (Especificar detalhes), especifique os nomes de pilha e chave:
  - a. Para Nome da pilha, digite **VPCE-Tutorial-Stack**.
  - b. Para KeyName, escolha VPCE-Tutorial-KeyPair.
  - c. Em SSHLocation, mantenha o valor padrão de **0.0.0.0/0**.

Specify Details

Specify a stack name and parameter values. You can use or change the default parameter values, which are defined in the AWS CloudFormation template. [Learn more.](#)

**Stack name**

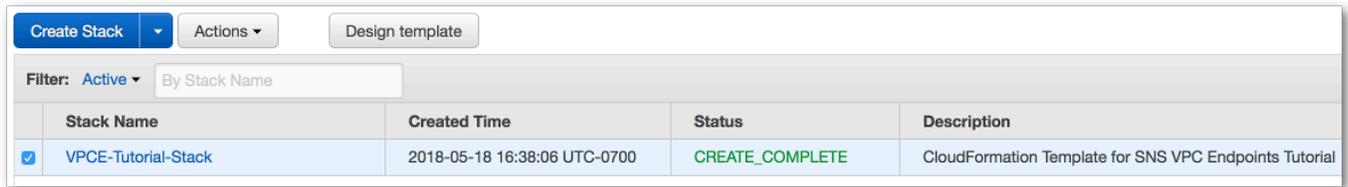
**Parameters**

**KeyName**  Name of an existing EC2 KeyPair to enable SSH access to the instance

**SSHLocation**  The IP address range that can be used to SSH to the EC2 instance

- d. Escolha Next (Próximo).
6. Na página Options (Opções), mantenha todos os valores padrão e escolha Next (Próximo).
  7. Na página Review (Revisar), verifique os detalhes da pilha.
  8. Em Capabilities (Recursos), confirme que o AWS CloudFormation pode criar recursos do IAM com nomes personalizados.
  9. Escolha Create (Criar).

O console do AWS CloudFormation abre a página Stacks (Pilhas). O VPCE-Tutorial-Stack tem um status de CREATE\_IN\_PROGRESS. Em instantes, após a conclusão do processo de criação, o status muda para CREATE\_COMPLETE.



Stack Name	Created Time	Status	Description
VPCE-Tutorial-Stack	2018-05-18 16:38:06 UTC-0700	CREATE_COMPLETE	CloudFormation Template for SNS VPC Endpoints Tutorial

### Tip

Escolha o botão Refresh (Atualizar) para ver o status mais recente da pilha.

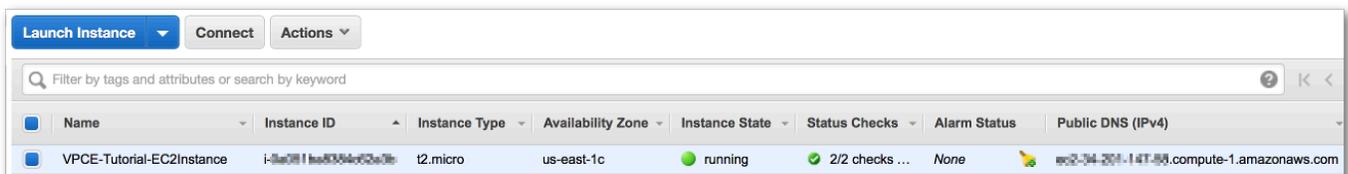
Etapa 3: verificar se a instância do Amazon EC2 não tem acesso à Internet

A instância do Amazon EC2 que foi lançada na sua VPC na etapa anterior não tem acesso à Internet. Ela não permite o tráfego de saída e não pode publicar mensagens no Amazon SNS. Para verificar isso, faça login na instância. Em seguida, conecte-se a um endpoint público e tente enviar uma mensagem Amazon SNS.

Neste ponto do tutorial, a tentativa de publicação falha. Em uma etapa posterior, depois de criar um VPC endpoint para o Amazon SNS, sua tentativa de publicação será bem-sucedida.

Para conectar-se à sua instância Amazon EC2.

1. Abra o console do Amazon EC2 em <https://console.aws.amazon.com/ec2/>.
2. No menu de navegação à esquerda, localize a seção Instances (Instâncias). Em seguida, selecione Instances (Instâncias).
3. Na lista de instâncias, selecione VPCE-Tutorial-EC2Instance.
4. Copie o nome do host fornecido na coluna Public DNS (IPv4) (DNS público [IPv4]).



Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS (IPv4)
VPCE-Tutorial-EC2Instance	i-081ba334c92e0b	t2.micro	us-east-1c	running	2/2 checks ...	None	ec2-34-204-147-85.compute-1.amazonaws.com

5. Abra um terminal. No diretório que contém o par de chaves, conecte-se à instância usando o seguinte comando, em que *instance-hostname* é o nome do host que você copiou do console do Amazon EC2:

```
$ ssh -i VPCE-Tutorial-KeyPair.pem ec2-user@instance-hostname
```

Para verificar se a instância não tem conectividade com a Internet

- No seu terminal, tente se conectar a qualquer endpoint público, como amazon.com:

```
$ ping amazon.com
```

Como a tentativa de conexão falha, você pode cancelar a qualquer momento (Ctrl + C no Windows ou Command + C no macOS).

Para verificar se a instância não tem conectividade com o Amazon SNS

1. Faça login no [console do Amazon SNS](#).
2. No menu de navegação à esquerda, escolha Topics (Tópicos).
3. Na página Topics (Tópicos), copie o nome de recurso da Amazon (ARN) para o tópico VPCE-Tutorial-Topic.
4. No seu terminal, tente publicar uma mensagem no tópico:

```
$ aws sns publish --region aws-region --topic-arn sns-topic-arn --message "Hello"
```

Como a tentativa de publicação falha, você pode cancelar a qualquer momento.

#### Etapa 4: criar um endpoint da VPC para Amazon SNS

Para conectar a VPC ao Amazon SNS, você define um VPC endpoint de interface. Depois de adicionar o endpoint, você poderá efetuar login na instância do Amazon EC2 na sua VPC. Lá, você poderá usar a API do Amazon SNS. Você pode publicar mensagens no tópico. Elas serão publicadas de maneira privada. As mensagens permanecerão na rede da AWS e não serão enviadas para a Internet pública.

**Note**

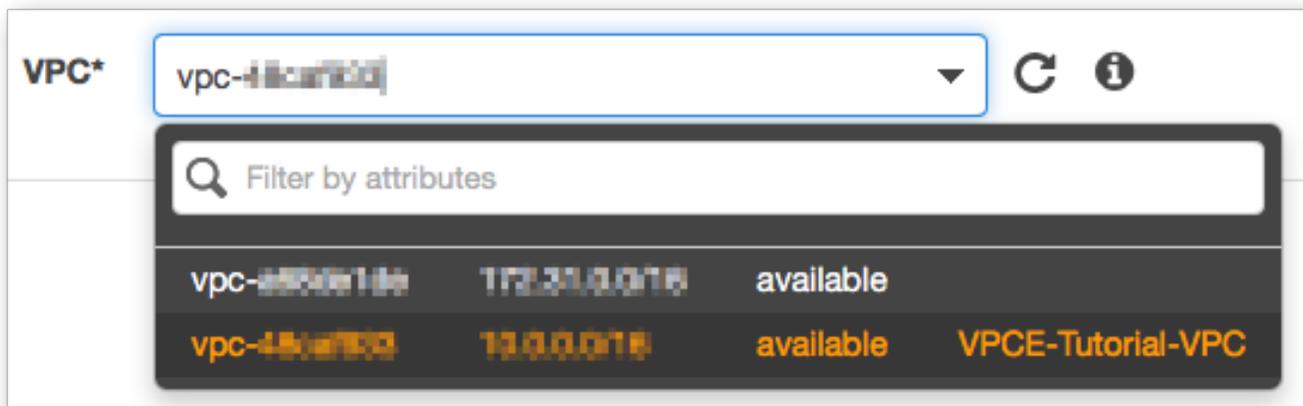
A instância ainda não tem acesso a outros produtos e endpoints da AWS na Internet.

Para criar o endpoint

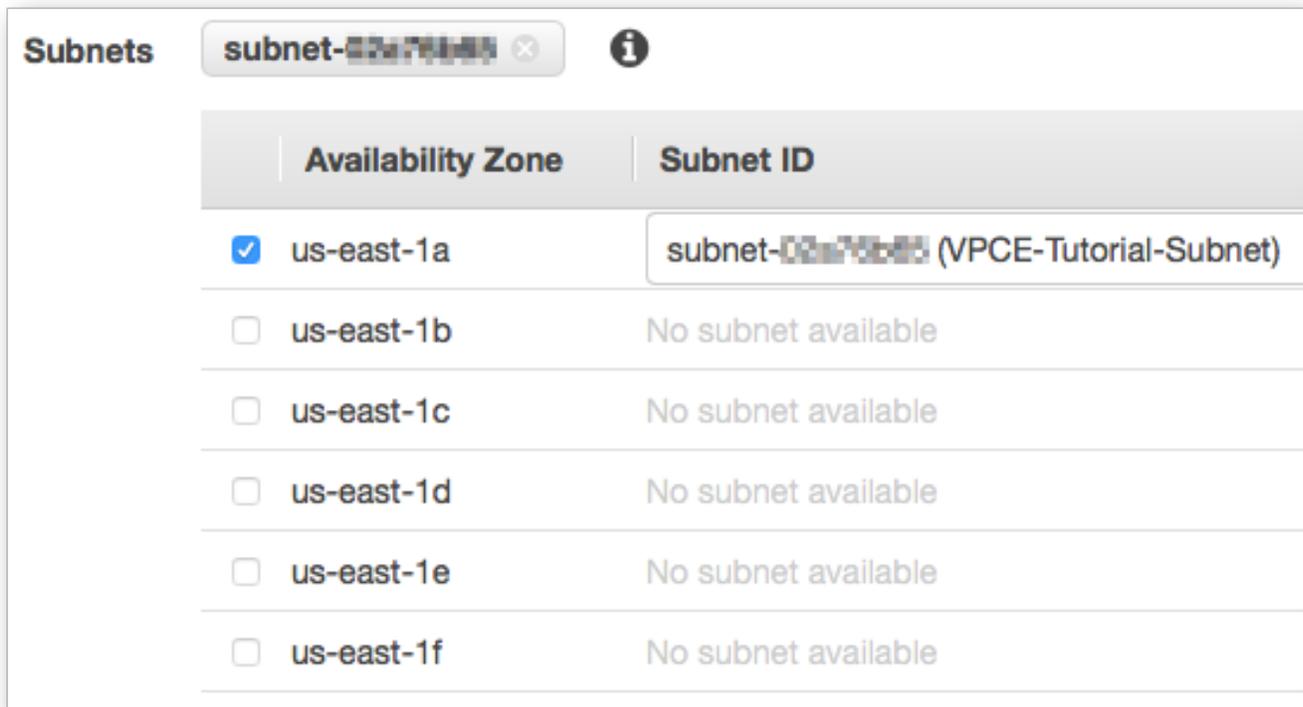
1. Abra o console da Amazon VPC em <https://console.aws.amazon.com/vpc/>.
2. No menu de navegação à esquerda, escolha Endpoints.
3. Escolha Create Endpoint.
4. Na página Create Endpoint (Criar endpoint), em Service category (Categoria de serviço), mantenha o valor padrão de AWS services (Serviços da AWS).
5. Em Service Name (Nome do serviço), escolha o nome do serviço para o Amazon SNS.

Os nomes de serviços variam de acordo com a região escolhida. Por exemplo, se você escolher Leste dos EUA (Norte da Virgínia), o nome do serviço será com.amazonaws.us-east-1.sns.

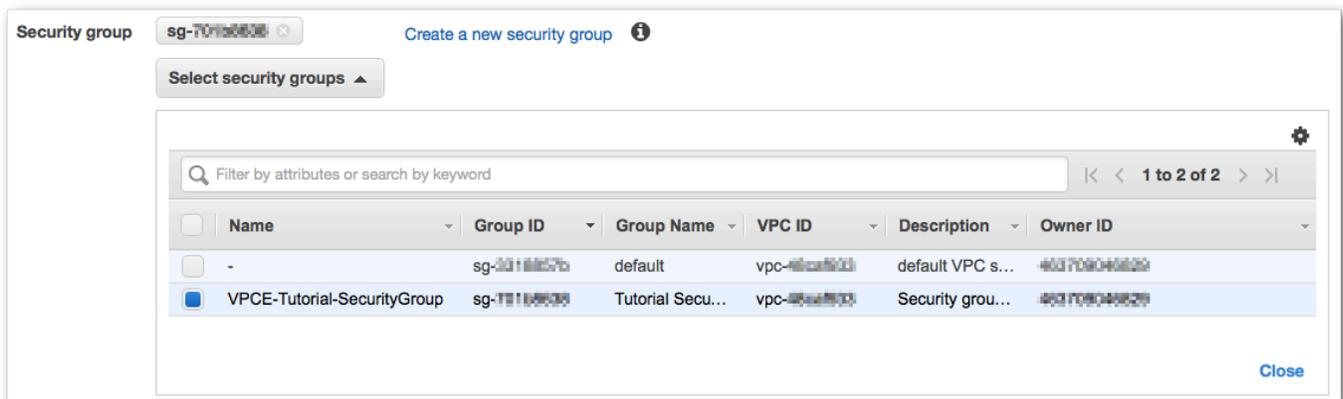
6. Para VPC, escolha a VPC com nome VPCE-Tutorial-VPC.



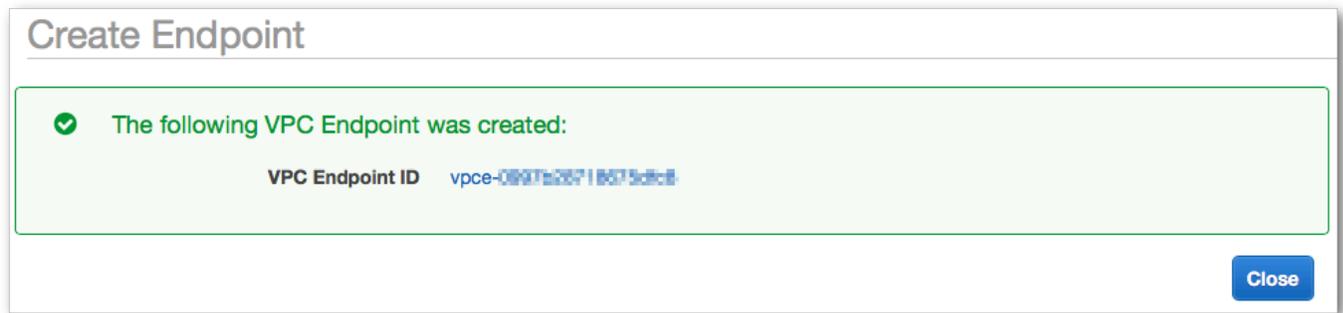
7. Em Subnets (Sub-redes), selecione aquela que contém VPCE-Tutorial-Subnet no ID de sub-rede.



8. Em Enable Private DNS Name (Ativar nome DNS privado), selecione Enable for this endpoint (Ativar para este terminal).
9. Em Security group (Grupo de segurança), escolha Select security group (Selecionar grupo de segurança) e VPCE-Tutorial-SecurityGroup.

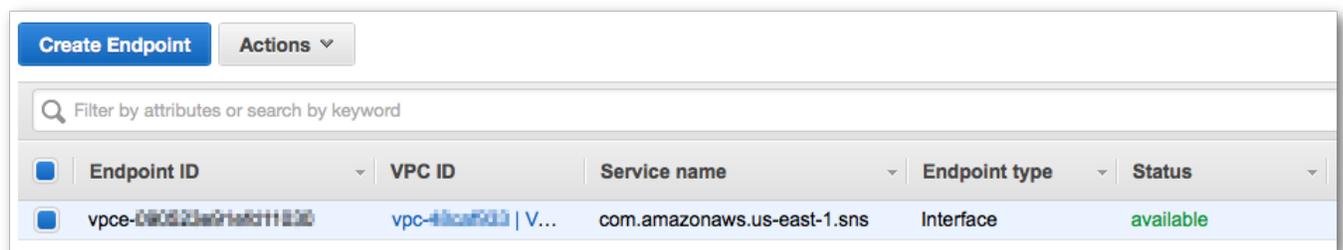


10. Escolha Create endpoint (Criar endpoint). O console da Amazon VPC verifica se o endpoint da VPC foi criado.



## 11. Escolha Close (Fechar).

O console da Amazon VPC abre a página Endpoints. O novo endpoint tem um status pending (pendente). Em instantes, após a conclusão do processo de criação, o status muda para available (disponível).



## Etapa 5: publicar uma mensagem no seu tópico do Amazon SNS

Agora que sua VPC inclui um endpoint para o Amazon SNS, você pode fazer login na instância do Amazon EC2 e publicar mensagens no tópico.

Para publicar uma mensagem

1. Se seu terminal não estiver mais conectado à sua instância do Amazon EC2, conecte-se novamente:

```
$ ssh -i VPCE-Tutorial-KeyPair.pem ec2-user@instance-hostname
```

2. Execute o mesmo comando que você fez anteriormente para publicar uma mensagem no seu tópico do Amazon SNS. Desta vez, a tentativa de publicação será bem-sucedida e o Amazon SNS retornará um ID de mensagem:

```
$ aws sns publish --region aws-region --topic-arn sns-topic-arn --message "Hello"
```

```
{  
  "MessageId": "5b111270-d169-5be6-9042-410dfc9e86de"  
}
```

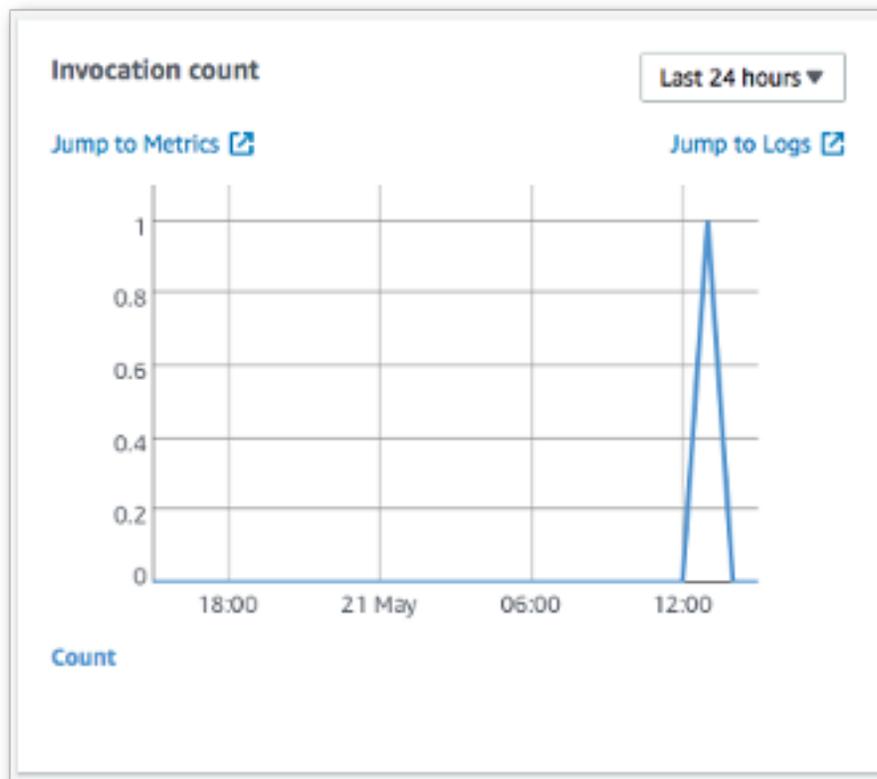
## Etapa 6: Verificar as entregas de mensagens

Quando o tópico do Amazon SNS receber uma mensagem, ele enviará a mensagem para as duas funções assinantes do Lambda. Quando essas funções receberem a mensagem, elas registrarão o evento nos logs do CloudWatch. Para saber se a entrega da mensagem foi bem sucedida, verifique se as funções foram invocadas e se os logs do CloudWatch foram atualizados.

Para verificar se as funções do Lambda foram invocadas

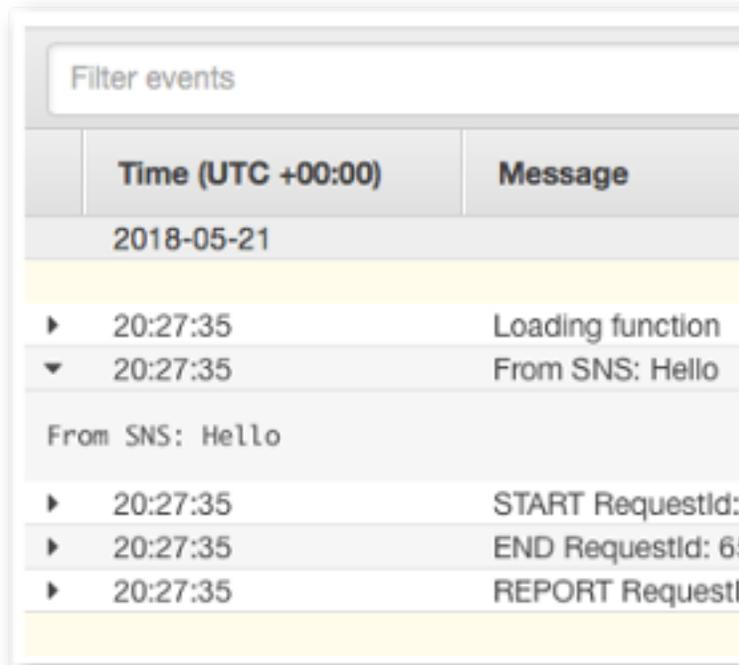
1. Abra o console do AWS Lambda em <https://console.aws.amazon.com/lambda/>.
2. Na página Functions (Funções), escolha VPCE-Tutorial-Lambda-1.
3. Escolha Monitoring.
4. Verifique o gráfico Invocation count (Contagem de invocação). Este gráfico mostra o número de vezes que a função do Lambda foi executada.

A contagem de invocação corresponde ao número de vezes que você publicou uma mensagem no tópico.



Para verificar se os logs do CloudWatch foram atualizados

1. Abra o console do CloudWatch em <https://console.aws.amazon.com/cloudwatch/>.
2. No menu de navegação à esquerda, escolha Logs.
3. Verifique os registros que foram escritos pelas funções do Lambda:
  - a. Escolha o grupo de logs `/aws/lambda/VPCE-Tutorial-Lambda-1/`.
  - b. Escolha o fluxo de logs.
  - c. Verifique se o log inclui a entrada `From SNS: Hello`.



	Time (UTC +00:00)	Message
2018-05-21		
▶	20:27:35	Loading function
▼	20:27:35	From SNS: Hello
From SNS: Hello		
▶	20:27:35	START RequestId:
▶	20:27:35	END RequestId: 65
▶	20:27:35	REPORT RequestId:

- d. Escolha Log Groups (Grupos de logs) na parte superior do console para retornar para a página Log Groups (Grupos de logs). Em seguida, repita as etapas anteriores para o grupo de log `/aws/lambda/VPCE-Tutorial-Lambda-2/`.

Parabéns! Ao adicionar um endpoint para o Amazon SNS a uma VPC, você conseguiu publicar uma mensagem em um tópico de dentro da rede gerenciada pela VPC. A mensagem foi publicada de maneira privada sem ser exposta à Internet pública.

### Etapa 7: limpeza

A menos que você queira manter os recursos criados, você poderá excluí-los agora. Excluindo os recursos da AWS que você não está mais usando, você evita cobranças desnecessárias em sua Conta da AWS.

Primeiro, exclua seu endpoint da VPC usando o console da Amazon VPC. Em seguida, exclua os outros recursos que você criou, excluindo a pilha no console do AWS CloudFormation. Quando você exclui uma pilha, o AWS CloudFormation remove os recursos da pilha de sua Conta da AWS.

### Para excluir seu VPC endpoint

1. Abra o console da Amazon VPC em <https://console.aws.amazon.com/vpc/>.
2. No menu de navegação à esquerda, escolha Endpoints.

3. Selecione o endpoint que você criou.
4. Escolha Actions (Ações) e escolha Delete Endpoint (Excluir endpoint).
5. Na janela Delete Endpoint (Excluir endpoint), escolha Yes, Delete (Sim, excluir).

O status do endpoint muda para deleting (excluindo). Quando a exclusão é concluída, o endpoint é removido da página.

Para excluir sua pilha do AWS CloudFormation

1. Abra o console do AWS CloudFormation em <https://console.aws.amazon.com/cloudformation>.
2. Selecione a pilha VPCE-Tutorial-Stack.
3. Escolha Ações e, em seguida, escolha Excluir pilha.
4. Na janela Delete Stack (Excluir pilha), escolha Yes, Delete (Sim, excluir).

O status da pilha muda para DELETE\_IN\_PROGRESS. Quando a exclusão é concluída, a pilha é removida da página.

Recursos relacionados

Para obter mais informações, consulte os recursos a seguir.

- [Blog de segurança da AWS: Proteção de mensagens publicadas no Amazon SNS com o AWS PrivateLink](#)
- [O que é Amazon VPC?](#)
- [VPC endpoints do](#)
- [O que é o Amazon EC2?](#)
- [AWS CloudFormation Conceitos do](#)

## Segurança de proteção de dados de mensagens

- [Proteção de dados de mensagens](#) é um recurso do Amazon SNS usado para definir regras e políticas próprias para auditar e controlar o conteúdo de dados em movimento, em vez de dados em repouso.
- A proteção de dados de mensagens oferece serviços de governança, conformidade e auditoria para aplicações corporativas centradas em mensagens, para que a entrada e a saída de dados

possam ser controladas pelo proprietário do tópico do Amazon SNS e os fluxos de conteúdo possam ser rastreados e registrados.

- Você pode criar regras de governança baseadas em carga útil para impedir a entrada de conteúdo não autorizado em seus fluxos de mensagens.
- Você pode conceder diferentes permissões de acesso ao conteúdo para assinantes individuais e auditar todo o processo de fluxo de conteúdo.

## Gerenciamento de identidade e acesso no Amazon SNS

O acesso ao Amazon SNS exige credenciais que a AWS possa usar para autenticar suas solicitações. Essas credenciais devem ter permissões para acessar recursos da AWS, como tópicos e mensagens do Amazon SNS. As seções a seguir fornecem detalhes sobre como é possível usar o [AWS Identity and Access Management \(IAM\)](#) e o Amazon SNS para ajudar a proteger seus recursos controlando quem pode acessá-los.

O AWS Identity and Access Management (IAM) é um AWS service (Serviço da AWS) que ajuda a controlar o acesso aos recursos da AWS de forma segura. Os administradores do IAM controlam quem pode ser autenticado (conectado) e autorizado (ter permissões) para utilizar os recursos do Amazon SNS. O IAM é um AWS service (Serviço da AWS) que pode ser usado sem custo adicional.

### Público

A forma de usar o AWS Identity and Access Management (IAM) varia em função do trabalho realizado no Amazon SNS.

**Usuário do serviço:** se você usar o serviço do Amazon SNS para fazer seu trabalho, o administrador fornecerá as credenciais e as permissões de que você precisa. À medida que mais recursos do Amazon SNS forem usados para realizar o trabalho, talvez sejam necessárias permissões adicionais. Entender como o acesso é gerenciado pode ajudá-lo a solicitar as permissões corretas ao seu administrador. Se você não puder acessar um recurso no Amazon SNS, consulte [Resolução de problemas de identidade e acesso do Amazon Simple Notification Service](#).

**Administrador do serviço:** se você for o responsável pelos recursos do Amazon SNS em sua empresa, provavelmente terá acesso total ao Amazon SNS. Cabe a você determinar quais funcionalidades e recursos do Amazon SNS os usuários do seu serviço devem acessar. Assim, você deve enviar solicitações ao administrador do IAM para alterar as permissões dos usuários de seu serviço. Revise as informações nesta página para entender os Introdução ao IAM. Para saber mais

sobre como a empresa pode usar o IAM com o Amazon SNS, consulte [Como o Amazon Simple Notification Service funciona com o IAM](#).

Administrador do IAM: se você é um administrador do IAM, talvez queira saber detalhes sobre como pode escrever políticas para gerenciar o acesso ao Amazon SNS. Para visualizar exemplos de políticas baseadas em identidade do Amazon SNS que podem ser usadas no IAM, consulte [Exemplos de políticas baseadas em identidade do Amazon Simple Notification Service](#).

## Autenticando com identidades

A autenticação é a forma como você faz login na AWS usando suas credenciais de identidade. É necessário ser autenticado (fazer login na AWS) como o usuário raiz da Conta da AWS, como usuário do IAM ou assumindo um perfil do IAM.

Você pode fazer login na AWS como uma identidade federada usando credenciais fornecidas por uma fonte de identidades. AWS IAM Identity Center Os usuários do IAM Identity Center, a autenticação única da empresa e as suas credenciais do Google ou do Facebook são exemplos de identidades federadas. Quando você faz login como uma identidade federada, o administrador já configurou anteriormente a federação de identidades usando perfis do IAM. Quando você acessa a AWS usando a federação, está indiretamente assumindo um perfil.

É possível fazer login no AWS Management Console ou no portal de acesso da AWS dependendo do tipo de usuário que você é. Para obter mais informações sobre como fazer login na AWS, consulte [Como fazer login na conta da Conta da AWS](#) no Guia do usuário do Início de Sessão da AWS.

Se você acessar a AWS programaticamente, a AWS fornecerá um kit de desenvolvimento de software (SDK) e uma interface da linha de comando (CLI) para você assinar criptograficamente as solicitações usando as suas credenciais. Se você não utilizar as ferramentas da AWS, deverá assinar as solicitações por conta própria. Para obter mais informações sobre como usar o método recomendado para assinar solicitações por conta própria, consulte [Assinar solicitações de API da AWS](#) no Guia do usuário do IAM.

Independentemente do método de autenticação usado, também pode ser exigido que você forneça mais informações de segurança. Por exemplo, a AWS recomenda o uso da autenticação multifator (MFA) para aumentar a segurança de sua conta. Para saber mais, consulte [Autenticação multifator](#) no Guia AWS IAM Identity Center do usuário. [Usar a autenticação multifator \(MFA\) na AWS](#) no Guia do usuário do IAM.

## Usuário root da Conta da AWS

Ao criar uma Conta da AWS, você começa com uma identidade de login que tem acesso completo a todos os recursos e Serviços da AWS na conta. Essa identidade, denominada usuário raiz da Conta da AWS, e é acessada por login com o endereço de e-mail e a senha que você usou para criar a conta. É altamente recomendável não usar o usuário raiz para tarefas diárias. Proteja as credenciais do usuário raiz e use-as para executar as tarefas que somente ele pode executar. Para obter a lista completa das tarefas que exigem login como usuário raiz, consulte [Tarefas que exigem credenciais de usuário raiz](#) no Guia do usuário do IAM.

## Identidade federada

Como prática recomendada, exija que os usuários, inclusive os que precisam de acesso de administrador, usem a federação com um provedor de identidades para acessar Serviços da AWS usando credenciais temporárias.

Identidade federada é um usuário de seu diretório de usuários corporativos, um provedor de identidades da web, o AWS Directory Service, o diretório do Centro de Identidade ou qualquer usuário que acesse os Serviços da AWS usando credenciais fornecidas por meio de uma fonte de identidade. Quando as identidades federadas acessam Contas da AWS, elas assumem funções que fornecem credenciais temporárias.

Para o gerenciamento de acesso centralizado, recomendamos usar o AWS IAM Identity Center. Você pode criar usuários e grupos no Centro de Identidade do IAM ou se conectar e sincronizar com um conjunto de usuários e grupos em sua própria fonte de identidade para uso em todas as suas Contas da AWS e aplicações. Para obter mais informações sobre o Centro de Identidade do IAM, consulte [O que é o Centro de Identidade do IAM?](#) no Guia do usuário do AWS IAM Identity Center.

## Grupos e usuários do IAM

Um [usuário do IAM](#) é uma identidade dentro da Conta da AWS que tem permissões específicas para uma única pessoa ou aplicação. Sempre que possível, recomendamos depender de credenciais temporárias em vez de criar usuários do IAM com credenciais de longo prazo, como senhas e chaves de acesso. No entanto, se você tiver casos de uso específicos que exijam credenciais de longo prazo com usuários do IAM, recomendamos alternar as chaves de acesso. Para obter mais informações, consulte [Altere as chaves de acesso regularmente para casos de uso que exijam credenciais de longo prazo](#) no Guia do usuário do IAM.

Um [grupo do IAM](#) é uma identidade que especifica uma coleção de usuários do IAM. Não é possível fazer login como um grupo. É possível usar grupos para especificar permissões para vários usuários

de uma vez. Os grupos facilitam o gerenciamento de permissões para grandes conjuntos de usuários. Por exemplo, você pode ter um grupo chamado IAMAdmins e atribuir a esse grupo permissões para administrar recursos do IAM.

Usuários são diferentes de funções. Um usuário é exclusivamente associado a uma pessoa ou a uma aplicação, mas uma função pode ser assumida por qualquer pessoa que precisar dela. Os usuários têm credenciais permanentes de longo prazo, mas as funções fornecem credenciais temporárias. Para saber mais, consulte [Quando criar um usuário do IAM \(em vez de uma função\)](#) no Guia do usuário do IAM.

## Funções do IAM

Um [perfil do IAM](#) é uma identidade dentro da Conta da AWS que tem permissões específicas. Ela é semelhante a um usuário do IAM, mas não está associada a uma pessoa específica. É possível assumir temporariamente um perfil do IAM no AWS Management Console [alternando perfis](#). É possível assumir um perfil chamando uma operação de API da AWS CLI ou da AWS, ou usando um URL personalizado. Para obter mais informações sobre métodos para o uso de perfis, consulte [Usar perfis do IAM](#) no Guia do usuário do IAM.

Os perfis do IAM com credenciais temporárias são úteis nas seguintes situações:

- **Acesso de usuário federado:** para atribuir permissões a identidades federadas, você pode criar um perfil e definir permissões para ele. Quando uma identidade federada é autenticada, essa identidade é associada ao perfil e recebe as permissões definidas pelo mesmo. Para obter mais informações sobre perfis para federação, consulte [Criar um perfil para um provedor de identidades de terceiros](#) no Guia do usuário do IAM. Se você usar o IAM Identity Center, configure um conjunto de permissões. Para controlar o que suas identidades podem acessar após a autenticação, o IAM Identity Center correlaciona o conjunto de permissões a um perfil no IAM. Para obter informações sobre conjuntos de permissões, consulte [Conjuntos de permissões](#) no Guia do usuário do AWS IAM Identity Center.
- **Permissões temporárias para usuários do IAM:** um usuário ou um perfil do IAM pode assumir um perfil do IAM para obter temporariamente permissões diferentes para uma tarefa específica.
- **Acesso entre contas:** é possível usar um perfil do IAM para permitir que alguém (uma entidade principal confiável) em outra conta acesse recursos em sua conta. As funções são a principal forma de conceder acesso entre contas. No entanto, alguns Serviços da AWS permitem que você anexe uma política diretamente a um recurso (em vez de usar uma função como proxy). Para saber a diferença entre funções e políticas baseadas em recurso para acesso entre contas,

consulte [Como as funções do IAM diferem das políticas baseadas em recurso](#) no Guia do usuário do IAM.

- Acesso entre serviços: alguns Serviços da AWS usam recursos em outros Serviços da AWS. Por exemplo, quando você faz uma chamada em um serviço, é comum que esse serviço execute aplicações no Amazon EC2 ou armazene objetos no Amazon S3. Um serviço pode fazer isso usando as permissões do principal de chamada, usando um perfil de serviço ou um perfil vinculado ao serviço.
- Encaminhamento de sessões de acesso (FAS): qualquer pessoa que utilizar uma função ou usuário do IAM para realizar ações na AWS é considerada uma entidade principal. Ao usar alguns serviços, você pode executar uma ação que inicia outra ação em um serviço diferente. O recurso FAS utiliza as permissões da entidade principal que chama um AWS service (Serviço da AWS), combinadas às permissões do AWS service (Serviço da AWS) solicitante, para realizar solicitações para serviços downstream. As solicitações de FAS só são feitas quando um serviço recebe uma solicitação que exige interações com outros Serviços da AWS ou com recursos para serem concluídas. Nesse caso, você precisa ter permissões para executar ambas as ações. Para obter detalhes da política ao fazer solicitações de FAS, consulte [Encaminhar sessões de acesso](#).
- Perfil de serviço: um perfil de serviço é um [perfil do IAM](#) que um serviço assume para realizar ações em seu nome. Um administrador do IAM pode criar, modificar e excluir um perfil de serviço do IAM. Para obter mais informações, consulte [Criar um perfil para delegar permissões a um AWS service \(Serviço da AWS\)](#) no Guia do usuário do IAM.
- Perfil vinculado ao serviço: um perfil vinculado ao serviço é um tipo de perfil de serviço vinculado a um AWS service (Serviço da AWS). O serviço pode assumir o perfil para executar uma ação em seu nome. Os perfis vinculados ao serviço aparecem em sua Conta da AWS e são de propriedade do serviço. Um administrador do IAM pode visualizar, mas não pode editar as permissões para perfis vinculados a serviço.
- Aplicações em execução no Amazon EC2: é possível usar um perfil do IAM para gerenciar credenciais temporárias para aplicações em execução em uma instância do EC2 e fazer solicitações da AWS CLI ou da AWS API. É preferível fazer isso e armazenar chaves de acesso na instância do EC2. Para atribuir uma função da AWS a uma instância do EC2 e disponibilizá-la para todas as suas aplicações, crie um perfil de instância que esteja anexado a ela. Um perfil de instância contém a função e permite que os programas em execução na instância do EC2 obtenham credenciais temporárias. Para mais informações, consulte [Usar um perfil do IAM para conceder permissões a aplicações em execução nas instâncias do Amazon EC2](#) no Guia do usuário do IAM.

Para saber se deseja usar os perfis do IAM, consulte [Quando criar um perfil do IAM \(em vez de um usuário\)](#) no Guia do usuário do IAM.

## Gerenciamento do acesso usando políticas

Você controla o acesso na AWS criando políticas e anexando-as a identidades ou recursos da AWS. Uma política é um objeto na AWS que, quando associado a uma identidade ou recurso, define suas permissões. A AWS avalia essas políticas quando uma entidade principal (usuário, usuário raiz ou sessão de função) faz uma solicitação. As permissões nas políticas determinam se a solicitação será permitida ou negada. A maioria das políticas são armazenadas na AWS como documentos JSON. Para obter mais informações sobre a estrutura e o conteúdo de documentos de políticas JSON, consulte [Visão geral das políticas JSON](#) no Guia do usuário do IAM.

Os administradores podem usar AWS as políticas JSON da para especificar quem tem acesso a quê. Ou seja, qual entidade principal pode executar ações em quais recursos e em que condições.

Por padrão, usuários e funções não têm permissões. Para conceder aos usuários permissão para executar ações nos recursos de que eles precisam, um administrador do IAM pode criar políticas do IAM. O administrador pode então adicionar as políticas do IAM a perfis, e os usuários podem assumir os perfis.

As políticas do IAM definem permissões para uma ação, independentemente do método usado para executar a operação. Por exemplo, suponha que você tenha uma política que permite a ação `iam:GetRole`. Um usuário com essa política pode obter informações de perfis do AWS Management Console, da AWS CLI ou da API da AWS.

### Políticas baseadas em identidade

As políticas baseadas em identidade são documentos de políticas de permissões JSON que você pode anexar a uma identidade, como usuário, grupo de usuários ou perfil do IAM. Essas políticas controlam quais ações os usuários e funções podem realizar, em quais recursos e em que condições. Para saber como criar uma política baseada em identidade, consulte [Criar políticas do IAM](#) no Guia do usuário do IAM.

As políticas baseadas em identidade podem ser categorizadas ainda mais como políticas em linha ou políticas gerenciadas. As políticas em linha são anexadas diretamente a um único usuário, grupo ou função. As políticas gerenciadas são políticas independentes que podem ser anexadas a vários usuários, grupos e funções na Conta da AWS. As políticas gerenciadas incluem políticas gerenciadas pela AWS e políticas gerenciadas pelo cliente. Para saber como escolher entre uma

política gerenciada ou uma política em linha, consulte [Escolher entre políticas gerenciadas e políticas em linha](#) no Guia do usuário do IAM.

## Políticas baseadas em recurso

Políticas baseadas em recurso são documentos de políticas JSON que você anexa a um recurso. São exemplos de políticas baseadas em recursos as políticas de confiança de perfil do IAM e as políticas de bucket do Amazon S3. Em serviços compatíveis com políticas baseadas em recursos, os administradores de serviço podem usá-las para controlar o acesso a um recurso específico. Para o recurso ao qual a política está anexada, a política define quais ações um principal especificado pode executar nesse recurso e em que condições. Você deve [especificar um principal](#) em uma política baseada em recursos. As entidades principais podem incluir contas, usuários, perfis, usuários federados ou Serviços da AWS.

Políticas baseadas em recursos são políticas em linha que estão localizadas nesse serviço. Não é possível usar as políticas gerenciadas da AWS do IAM em uma política baseada em recursos.

## Listas de controle de acesso (ACLs)

As listas de controle de acesso (ACLs) controlam quais entidades principais (membros, usuários ou funções da conta) têm permissões para acessar um recurso. As ACLs são semelhantes às políticas baseadas em recursos, embora não usem o formato de documento de política JSON.

Amazon S3, AWS WAF e Amazon VPC são exemplos de serviços que oferecem suporte a ACLs. Para saber mais sobre ACLs, consulte [Visão geral da lista de controle de acesso \(ACL\)](#) no Guia do desenvolvedor do Amazon Simple Storage Service.

## Outros tipos de política

A AWS aceita tipos de política menos comuns. Esses tipos de política podem definir o máximo de permissões concedidas a você pelos tipos de política mais comuns.

- **Limites de permissões:** um limite de permissões é um recurso avançado no qual você define o máximo de permissões que uma política baseada em identidade pode conceder a uma entidade do IAM (usuário ou perfil do IAM). É possível definir um limite de permissões para uma entidade. As permissões resultantes são a interseção das políticas baseadas em identidade de uma entidade e dos seus limites de permissões. As políticas baseadas em recurso que especificam o usuário ou a função no campo `Principal` não são limitadas pelo limite de permissões. Uma negação explícita em qualquer uma dessas políticas substitui a permissão. Para obter mais

informações sobre limites de permissões, consulte [Limites de permissões para identidades do IAM](#) no Guia do usuário do IAM.

- **Políticas de controle de serviço (SCPs):** SCPs são políticas JSON que especificam as permissões máximas para uma organização ou unidade organizacional (UO) no AWS Organizations. O AWS Organizations é um serviço para agrupar e gerenciar centralmente várias Contas da AWS pertencentes à sua empresa. Se você habilitar todos os recursos em uma organização, poderá aplicar políticas de controle de serviço (SCPs) a qualquer uma ou a todas as contas. Uma SCP limita as permissões para entidades em contas-membro, inclusive para cada usuário raiz da Conta da AWS. Para obter mais informações sobre o Organizations e SCPs, consulte [Como os SCPs funcionam](#) no Guia do usuário do AWS Organizations.
- **Políticas de sessão:** são políticas avançadas que você transmite como um parâmetro quando cria de forma programática uma sessão temporária para uma função ou um usuário federado. As permissões da sessão resultante são a interseção das políticas baseadas em identidade do usuário ou da função e das políticas de sessão. As permissões também podem ser provenientes de uma política baseada em recurso. Uma negação explícita em qualquer uma dessas políticas substitui a permissão. Para obter mais informações, consulte [Políticas de sessão](#) no Guia do usuário do IAM.

## Vários tipos de política

Quando vários tipos de política são aplicáveis a uma solicitação, é mais complicado compreender as permissões resultantes. Para saber como a AWS determina se deve permitir uma solicitação quando há vários tipos de política envolvidos, consulte [Lógica da avaliação](#) de políticas no Guia do usuário do IAM.

## Controle de acesso

O Amazon SNS tem seu próprio sistema de permissões com base em recursos que usa políticas escritas na mesma linguagem usada para políticas do AWS Identity and Access Management (IAM). Isso significa que você pode atingir objetivos semelhantes com as políticas do IAM e do Amazon SNS.

### Note

É importante entender que todas as Contas da AWS podem delegar suas permissões para usuários em suas contas. O acesso entre contas permite compartilhar o acesso a seus recursos da AWS sem a necessidade de gerenciar usuários adicionais. Para obter

informações sobre como usar o acesso entre contas, consulte [Habilitar o acesso entre contas](#) no Guia do usuário do IAM.

## Visão geral do gerenciamento de acesso no Amazon SNS

Esta seção descreve os conceitos básicos que você precisa conhecer para usar a linguagem de políticas de acesso para criar políticas. Ela também descreve o processo geral de como o controle de acesso funciona com a linguagem de políticas de acesso e como as políticas são avaliadas.

### Tópicos

- [Quando usar o controle de acesso](#)
- [Principais conceitos](#)
- [Visão geral da arquitetura](#)
- [Utilização da linguagem de políticas de acesso](#)
- [Lógica de avaliação](#)
- [Casos de exemplo para controle de acesso do Amazon SNS](#)

### Quando usar o controle de acesso

Você tem muita flexibilidade com relação à forma de conceder ou negar acesso a um recurso. No entanto, os casos de uso típicos são bastante simples:

- Você deseja conceder a outra Conta da AWS determinado tipo de ação de tópico (por exemplo, Publicação). Para ter mais informações, consulte [Conceder acesso a um tópico a uma Conta da AWS](#).
- Você deseja limitar as assinaturas do tópico apenas ao protocolo HTTPS. Para ter mais informações, consulte [Limitar assinaturas para HTTPS](#).
- Você deseja permitir que o Amazon SNS publique mensagens para a fila do Amazon SQS. Para ter mais informações, consulte [Publicar mensagens em uma fila do Amazon SQS](#).

### Principais conceitos

As seções a seguir descrevem os conceitos que você precisa entender para usar a linguagem de políticas de acesso. Eles são apresentados em uma ordem lógica, com os primeiros termos que você precisa saber na parte superior da lista.

## Tópicos

- [Permissão](#)
- [Statement](#)
- [Política](#)
- [Emissor](#)
- [Entidade principal](#)
- [Ação](#)
- [Recurso](#)
- [Condições e chaves](#)
- [Solicitante](#)
- [Avaliação](#)
- [Efeito](#)
- [Negação padrão](#)
- [Permitir](#)
- [Negação explícita](#)

## Permissão

Permissão refere-se ao conceito de conceder ou negar algum tipo de acesso a um recurso específico. Basicamente, as permissões seguem este formato: “A tem/não tem permissão para realizar B para C, quando D se aplicar”. Por exemplo, Jane (A) tem permissão para publicar (B) em TopicA (C) desde que ela utilize o protocolo HTTP (D). Sempre que Jane fizer uma publicação em TopicA, o serviço verificará se ela tem permissão e se a solicitação está de acordo com as condições definidas na permissão.

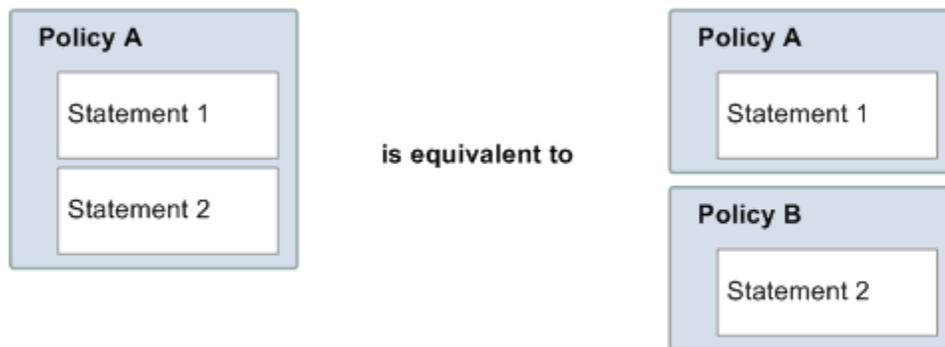
## Statement

Declaração é a descrição formal de uma única permissão, criada na linguagem de políticas de acesso. Você sempre cria uma declaração como parte de um documento de contêiner mais abrangente conhecido como política (consulte o próximo conceito).

## Política

Política é um documento (escrito na linguagem de políticas de acesso) que funciona como um contêiner para uma ou mais declarações. Por exemplo, uma política poderia ter duas declarações: uma que afirme que Jane pode se inscrever usando o protocolo de e-mail, e outra afirmando que

Bob não pode fazer uma publicação no tópico A. Como mostrado na figura a seguir, um cenário equivalente seria ter duas políticas, uma que afirme que Jane pode se inscrever usando o protocolo de e-mail e outra afirmando que Bob não pode publicar no tópico A.



Somente caracteres ASCII são permitidos em documentos de política. Você pode utilizar `aws:SourceAccount` e `aws:SourceOwner` para contornar o cenário em que você precisa conectar outros ARNs de serviços da AWS que contenham caracteres que não sejam ASCII. Veja a diferença entre [aws:SourceAccount versus aws:SourceOwner](#).

## Emissor

Emissor é a pessoa que cria uma política para conceder permissões a um recurso. Por definição, o emissor é sempre o proprietário do recurso. A AWS não permite que usuários de serviços da AWS criem políticas para recursos dos quais não são proprietários. Se João for o proprietário do recurso, a AWS autenticará sua identidade quando ele enviar a política que criou para conceder permissões a esse recurso.

## Entidade principal

Entidade principal é a pessoa ou as pessoas que recebem a permissão na política. A entidade principal é A na declaração “A tem permissão para realizar B para C, quando D se aplicar”. Em uma política, você pode definir a entidade principal como “ninguém” (ou seja, pode especificar um curinga para representar todas as pessoas). Você poderá fazer isso, por exemplo, se não quiser restringir o acesso com base na identidade real do solicitante, mas em outras características de identificação, como o endereço IP do solicitante.

## Ação

Ação é a atividade que a entidade principal tem permissão para executar. A ação é B na declaração “A tem permissão para realizar B para C, quando D se aplicar”. Normalmente, a ação é apenas a

operação na solicitação à AWS. Por exemplo, Jane envia uma solicitação ao Amazon SNS com `Action=Subscribe`. Você pode especificar uma ou várias ações em uma política.

## Recurso

Recurso é o objeto ao qual a entidade principal está solicitando acesso. O recurso é C na declaração “A tem permissão para realizar B para C, quando D se aplicar”.

## Condições e chaves

Condições são todas as restrições ou detalhes sobre a permissão. A condição é D na declaração “A tem permissão para realizar B para C, quando D se aplicar”. A parte da política que especifica as condições pode ser a mais detalhada e complexa de todas as partes. As condições comuns são relacionadas a:

- Data e hora (por exemplo, a solicitação precisa chegar antes de um dia específico)
- Endereço IP (por exemplo, o endereço IP do solicitante precisa ser fazer parte de determinado intervalo CIDR)

Chave é a característica específica que fundamenta a restrição de acesso. Por exemplo, a data e a hora da solicitação.

Você usa condições e chaves em conjunto para expressar a restrição. A maneira mais fácil de entender como você realmente implementa uma restrição é com um exemplo: se você deseja restringir o acesso para antes de 30 de maio de 2010, deve usar a condição chamada `DateLessThan`. Você usa a chave chamada `aws:CurrentTime` e a define com o valor `2010-05-30T00:00:00Z`. A AWS define as condições e chaves que você pode usar. O próprio serviço da AWS (por exemplo, Amazon SQS ou Amazon SNS) também pode definir as chaves específicas do serviço. Para ter mais informações, consulte [Permissões da API do Amazon SNS: referência de ações e recursos](#).

## Solicitante

Solicitante é a pessoa que envia uma solicitação a um serviço da AWS e pede acesso a um recurso específico. O solicitante envia uma solicitação à AWS que essencialmente pergunta: “Posso fazer B para C quando D se aplicar?”.

## Avaliação

Avaliação é o processo que o serviço da AWS usa para determinar se uma solicitação recebida deve ser negada ou permitida com base nas políticas aplicáveis. Para obter informações sobre a lógica de avaliação, consulte [Lógica de avaliação](#).

### Efeito

Efeito é o resultado que você deseja que uma declaração de política retorne no tempo da avaliação. Você especifica esse valor ao criar as declarações em uma política, e os valores possíveis são negar e permitir.

Por exemplo, você pode criar uma política que tenha uma declaração que nega todas as solicitações originárias da Antártica (efeito=negar considerando que a solicitação usa um endereço IP alocado para a Antártica). Como alternativa, você pode escrever uma política que tenha uma declaração que permita todas as solicitações que não são originárias da Antártica (effect=allow, considerando que a solicitação não é originária da Antártica). Embora as duas declarações pareçam realizar a mesma ação, na lógica da linguagem de políticas de acesso, elas são diferentes. Para ter mais informações, consulte [Lógica de avaliação](#).

Embora haja apenas dois valores possíveis que você pode especificar para o efeito (permitir ou negar), pode haver três resultados diferentes no tempo de avaliação da política: negação padrão, permitir ou negação explícita. Para obter mais informações, consulte os seguintes conceitos e [Lógica de avaliação](#).

### Negação padrão

Negação padrão é o resultado padrão de uma política na ausência de permitir ou negação explícita.

### Permitir

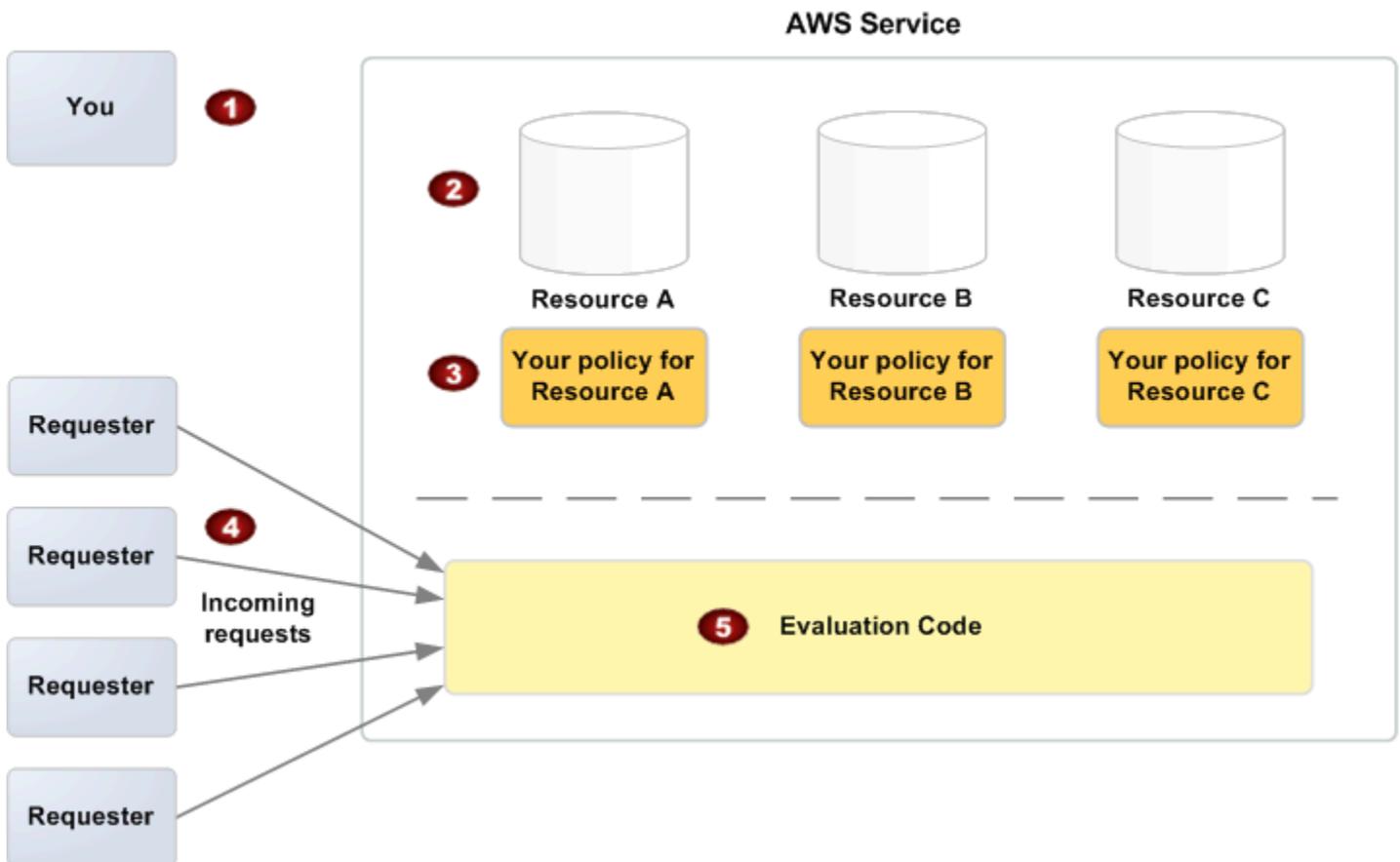
Permitir resulta de uma declaração que tem efeito = permitir, supondo que todas as condições indicadas foram atendidas. Exemplo: permitir solicitações se elas forem recebidas antes de 13h00 em 30 de abril de 2010. Um permitir substitui todas as negações padrão, mas nunca uma negação explícita.

### Negação explícita

A negação explícita resulta de uma declaração que tem efeito = negar, supondo que todas as condições indicadas foram atendidas. Exemplo: negar todas as solicitações originárias da Antártica. Todas as solicitações originárias da Antártica sempre serão negadas independentemente do que qualquer outra política possa permitir.

## Visão geral da arquitetura

A figura e a tabela a seguir descrevem os principais componentes que interagem para fornecer controle de acesso aos seus recursos.



1 Você, o proprietário do recurso.

2 Seus recursos contidos no serviço da AWS (por exemplo, filas do Amazon SQS).

3 Suas políticas.

Normalmente, você tem uma única política por recurso, embora seja possível ter várias. O próprio serviço da AWS fornece uma API que você usa para carregar e gerenciar suas políticas.

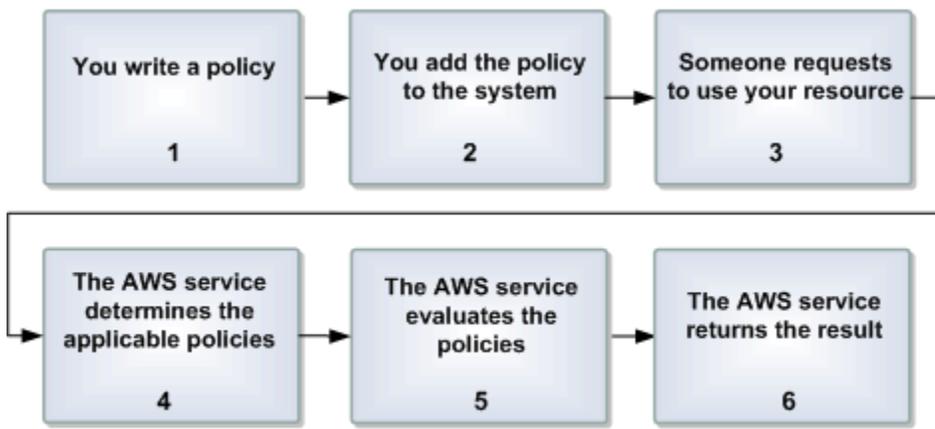
4 Os solicitantes e as respectivas solicitações de entrada para o serviço da AWS.

5 O código de avaliação da linguagem de políticas de acesso.

Esse é o conjunto de códigos dentro do serviço da AWS que avalia as solicitações de entrada em relação às políticas aplicáveis e determina se o solicitante tem permissão para acessar o recurso. Para obter informações sobre como o serviço toma decisões, consulte [Lógica de avaliação](#).

## Utilização da linguagem de políticas de acesso

A figura e a tabela a seguir descrevem o processo geral de como o controle de acesso funciona com a linguagem de políticas de acesso.



### Processo para usar o controle de acesso com a linguagem de políticas de acesso

- 1 Você cria uma política para seu recurso.

Por exemplo, você pode criar uma política para especificar permissões para seus tópicos do Amazon SNS.

- 2 Você faz upload da política na AWS.

O próprio serviço da AWS fornece uma API que você usa para carregar suas políticas. Por exemplo, você pode usar a ação `SetTopicAttributes` do Amazon SNS com a finalidade de carregar uma política para determinado tópico do Amazon SNS.

- 3 Alguém envia uma solicitação para usar seu recurso.

Por exemplo, um usuário envia uma solicitação ao Amazon SNS para usar um de seus tópicos.

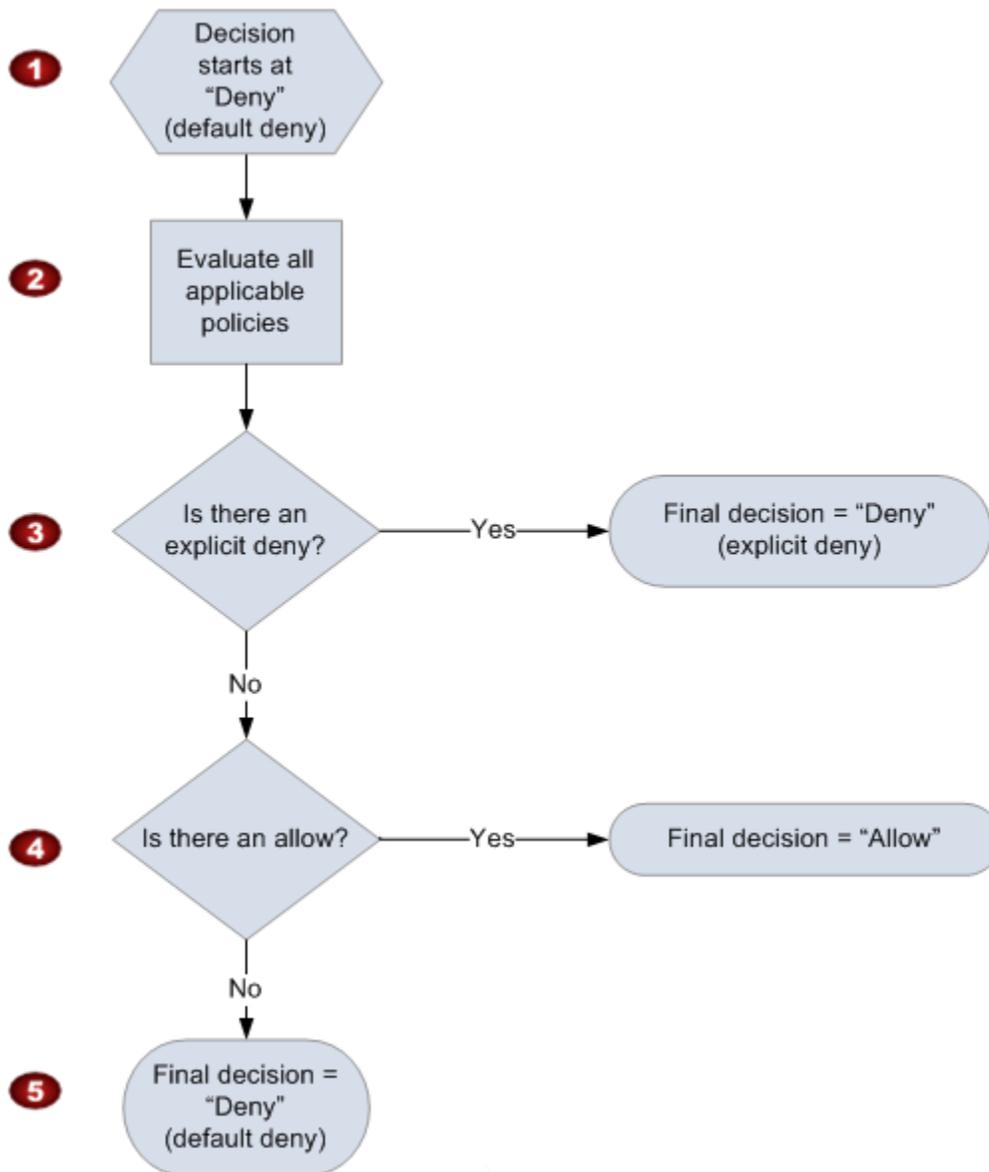
- 4 O serviço da AWS determina quais políticas são aplicáveis à solicitação.  
Por exemplo, o Amazon SNS verifica todas as políticas disponíveis do Amazon SNS e determina quais são aplicáveis (com base no recurso, no solicitante etc.).
- 5 O serviço da AWS avalia as políticas.  
Por exemplo, o Amazon SNS avalia as políticas e determina se o solicitante tem permissão para usar o tópico ou não. Para obter informações sobre a lógica de decisão, consulte [Lógica de avaliação](#).
- 6 O serviço da AWS nega a solicitação ou continua a processá-la.  
Por exemplo, com base no resultado de avaliação da política, o serviço retorna um erro de “acesso negado” para o solicitante ou continua a processar a solicitação.

## Lógica de avaliação

No momento da avaliação, o objetivo é decidir se uma solicitação de concessão deve ser permitida ou negada. A lógica de avaliação segue várias regras básicas:

- Por padrão, todas as solicitações para usar o recurso que venham de outras pessoas, e não de você, serão negadas.
- Um valor permitir substitui qualquer negação padrão.
- Uma negação explícita substitui todas as permissões.
- A ordem em que as políticas são avaliadas não é importante.

O seguinte fluxograma e discussão descrevem em mais detalhes como a decisão é tomada.



- 1 A decisão começa com uma negação padrão.
- 2 O código de aplicação avalia todas as políticas aplicáveis à solicitação (com base no recurso, na entidade principal, na ação e nas condições).  
A ordem em que o código de aplicação avalia as políticas não é importante.
- 3 Em todas essas políticas, o código de aplicação procura uma instrução de negação explícita que se aplica à solicitação.

Se o código de aplicação encontrar uma instrução desse tipo, ele retornará a decisão de “negar”, e o processo será concluído (essa é uma negação explícita; para obter mais informações, consulte [Negação explícita](#)).

4 Se nenhuma negação explícita for encontrada, o código de aplicação procurará qualquer instrução “permitir” aplicável à solicitação.

Se encontrar uma instrução “permitir”, o código de aplicação retornará a decisão de “permitir”, e o processo será concluído (o serviço continuará a processar a solicitação).

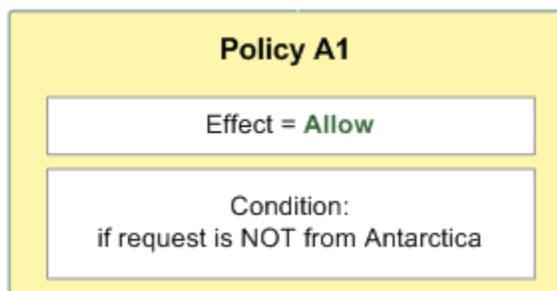
5 Se nenhuma instrução de permitir for encontrada, a decisão final será “negar”, porque não havia negação explícita, ou permitir, porque é considerada uma negação padrão (para obter mais informações, consulte [Negação padrão](#)).

### Interação entre negações explícitas e padrão

Uma política resulta em uma negação padrão caso não se aplique diretamente à solicitação. Por exemplo, se a solicitação do usuário for para usar o Amazon SNS, mas a política sobre o tópico não se referir à Conta da AWS do usuário, essa política resultará em uma negação padrão.

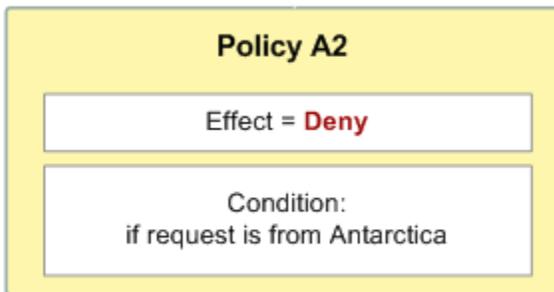
Uma política também resultará em uma negação padrão se uma condição em uma declaração não for atendida. Se todas as condições da declaração forem atendidas, a política resultará em permitir ou negação explícita, com base no valor do elemento efeito na política. As políticas não especificam o que fazer se uma condição não for atendida, e, portanto, o resultado padrão nesse caso é uma negação padrão.

Por exemplo, digamos que você deseja impedir solicitações que venham da Antártica. Você cria uma política (chamada Política A1) que permite uma solicitação somente se ela não for originária da Antártica. O seguinte diagrama ilustra a política.



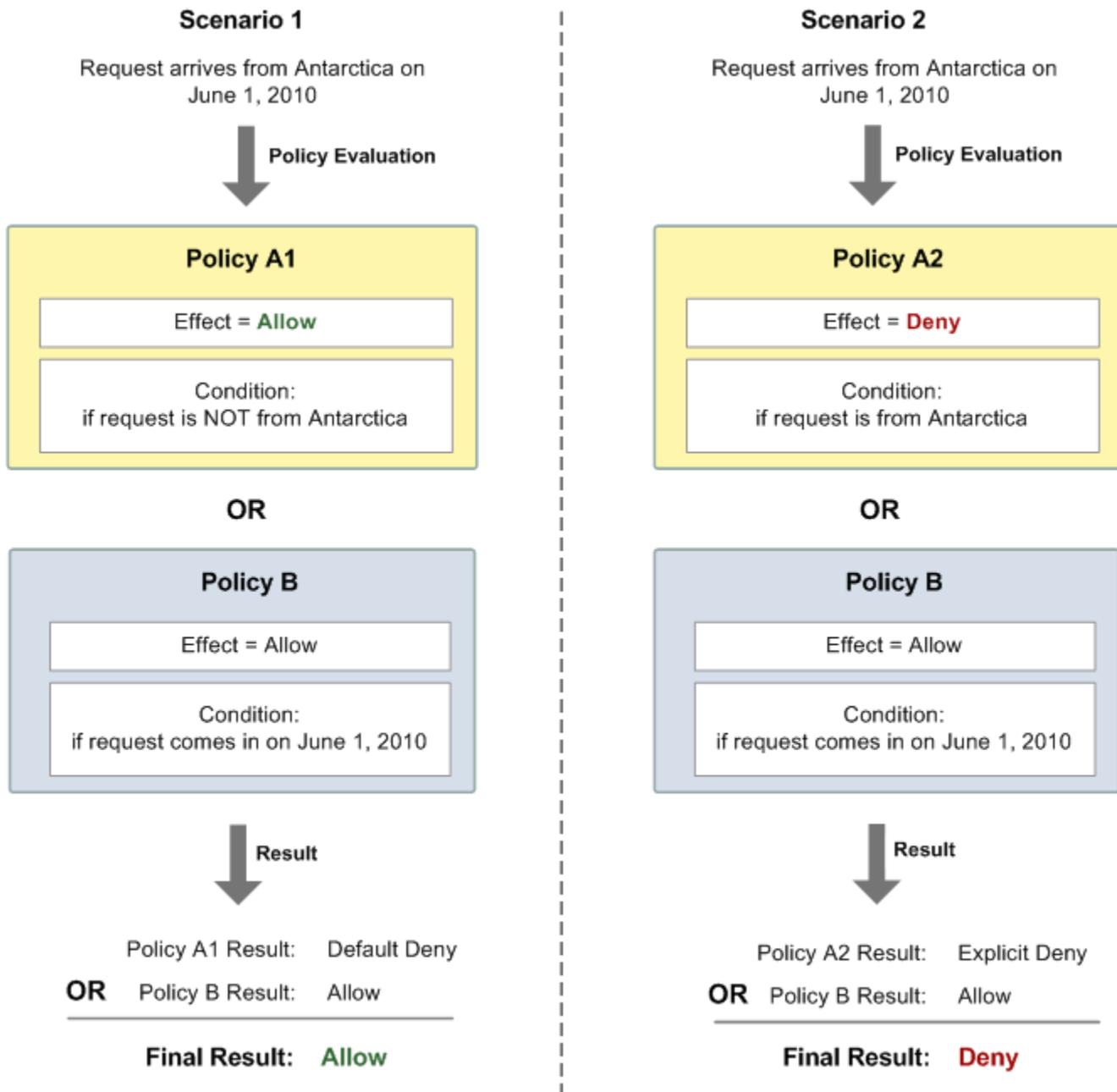
Se alguém enviar uma solicitação dos EUA, a condição será atendida (a solicitação não é originária da Antártica). Portanto, a solicitação será permitida. Porém, se alguém enviar uma solicitação da Antártica, a condição não será atendida, e, assim, o resultado da política será uma negação padrão.

Você pode transformar o resultado em uma negação explícita recriando a política (chamada Política A2), como mostrado no diagrama a seguir. Aqui, a política negará explicitamente uma solicitação se ela for originária da Antártica.



Se alguém enviar uma solicitação da Antártica, a condição será atendida, e o resultado da política será uma negação explícita.

A distinção entre negação padrão e negação explícita é importante porque uma negação padrão pode ser substituída por uma permissão, mas uma negação explícita não pode. Por exemplo, digamos que haja outra política que permita solicitações se elas chegarem em 1.º de junho de 2010. Como essa política afeta o resultado geral quando associada à política que restringe o acesso da Antártica? Compararemos o resultado geral ao associar a política baseada em data (que chamaremos de Política B) com as políticas anteriores A1 e A2. O Cenário 1 associa a Política A1 à Política B, e o Cenário 2 associa a Política A2 à Política B. A figura e a discussão a seguir mostram os resultados quando uma solicitação chega da Antártica em 1.º de junho de 2010.



No Cenário 1, a Política A1 retorna uma negação padrão, conforme descrito anteriormente nesta seção. A Política B retorna um permitir porque a política (por definição) permite solicitações que chegam em 1.º de junho de 2010. O permitir da Política B substitui a negação padrão da Política A1 e, assim, a solicitação é permitida.

No Cenário 2, a Política A2 retorna uma negação explícita, conforme descrito anteriormente nesta seção. Novamente, a Política B retorna um permitir. A negação explícita da Política A2 substitui o permitir da Política B e, assim, a solicitação é negada.

## Casos de exemplo para controle de acesso do Amazon SNS

Esta seção descreve alguns exemplos de casos de uso comuns para controle de acesso.

### Tópicos

- [Conceder acesso a um tópico a uma Conta da AWS](#)
- [Limitar assinaturas para HTTPS](#)
- [Publicar mensagens em uma fila do Amazon SQS](#)
- [Permitir que notificações de eventos do Simple Storage Service \(Amazon S3\) publiquem em um tópico](#)
- [Permitir que o Amazon SES publique em um tópico pertencente a outra conta](#)
- [aws:SourceAccount versus aws:SourceOwner](#)
- [Permitir que contas em uma organização no AWS Organizations publiquem em um tópico em uma conta diferente](#)
- [Permitir que qualquer CloudWatch alarme seja publicado em um tópico em uma conta diferente](#)
- [Restringir a publicação em um tópico do Amazon SNS somente de um endpoint da VPC específico](#)

### Conceder acesso a um tópico a uma Conta da AWS

Digamos que você tenha um tópico no sistema do Amazon SNS. No caso mais simples, você deseja permitir o acesso de uma ou mais Contas da AWS a determinada ação de tópico (por exemplo, Publicação).

Você pode fazer isso usando a ação `AddPermission` da API do Amazon SNS. É necessário um tópico, uma lista de IDs de Conta da AWS, uma lista de ações e um rótulo para criar automaticamente uma nova declaração na política de controle de acesso do tópico. Nesse caso, você não cria uma política por conta própria, porque o Amazon SNS gera automaticamente a nova declaração de política para você. Você pode remover a declaração de política posteriormente chamando `RemovePermission` com o respectivo rótulo.

Por exemplo, se você `AddPermission` acessasse o tópico `arn:aws:sns:us-east-2:444455556666:`, com o MyTopic ID `1111-2222-3333`, a ação e o rótulo Conta da AWS, o Amazon SNS geraria e inseriria a seguinte declaração de política de controle de acesso: `Publish grant-1234-publish`

```
{
```

```
"Statement": [{
  "Sid": "grant-1234-publish",
  "Effect": "Allow",
  "Principal": {
    "AWS": "111122223333"
  },
  "Action": ["sns:Publish"],
  "Resource": "arn:aws:sns:us-east-2:444455556666:MyTopic"
}]
}
```

Assim que essa declaração for adicionada, o usuário com a Conta da AWS 1111-2222-3333 poderá publicar mensagens no tópico.

### Limitar assinaturas para HTTPS

No exemplo a seguir, limite o protocolo de entrega de notificações para HTTPS.

Você precisa saber como criar sua própria política para o tópico porque a ação `AddPermission` do Amazon SNS não permite que você especifique uma restrição de protocolo ao conceder acesso a seu tópico a alguém. Nesse caso, você pode criar sua própria política e usar a ação `SetTopicAttributes` para definir o atributo `Policy` do tópico para a nova política.

O exemplo a seguir de uma política completa concede ao ID da Conta da AWS 1111-2222-3333 a capacidade de assinar as notificações de um tópico.

```
{
  "Statement": [{
    "Sid": "Statement1",
    "Effect": "Allow",
    "Principal": {
      "AWS": "111122223333"
    },
    "Action": ["sns:Subscribe"],
    "Resource": "arn:aws:sns:us-east-2:444455556666:MyTopic",
    "Condition": {
      "StringEquals": {
        "sns:Protocol": "https"
      }
    }
  }]
}
```

## Publicar mensagens em uma fila do Amazon SQS

Nesse caso de uso, você deseja publicar mensagens de seu tópico em sua fila do Amazon SQS. Como o Amazon SNS, o Amazon SQS usa a linguagem de política de controle de acesso da Amazon. Para permitir que o Amazon SNS envie mensagens, você precisa usar a ação `SetQueueAttributes` do Amazon SQS para definir uma política na fila.

Novamente, você precisa saber como criar sua própria política porque a ação `AddPermission` do Amazon SQS não cria declarações de política com condições.

### Note

O exemplo apresentado abaixo é uma política do Amazon SQS (controle de acesso à sua fila), não uma política do Amazon SNS (controle de acesso ao seu tópico). As ações são ações do Amazon SQS, e o recurso é o nome do recurso da Amazon (ARN) da fila. Você pode determinar o ARN da fila recuperando o atributo `QueueArn` da fila com a ação `GetQueueAttributes`.

```
{
  "Statement": [{
    "Sid": "Allow-SNS-SendMessage",
    "Effect": "Allow",
    "Principal": {
      "Service": "sns.amazonaws.com"
    },
    "Action": ["sqs:SendMessage"],
    "Resource": "arn:aws:sqs:us-east-2:444455556666:MyQueue",
    "Condition": {
      "ArnEquals": {
        "aws:SourceArn": "arn:aws:sns:us-east-2:444455556666:MyTopic"
      }
    }
  }]
}
```

Essa política usa a condição `aws:SourceArn` para restringir o acesso à fila com base na origem da mensagem que está sendo enviada à fila. Você pode usar esse tipo de política para permitir que o Amazon SNS envie mensagens à fila somente se as mensagens forem provenientes de um de seus

próprios tópicos. Nesse caso, você especifica um tópico específico, cujo ARN é `arn:aws:sns:us-east-2:444455556666:MyTopic`.

A política anterior é um exemplo da política do Amazon SQS que você pode criar e adicionar a uma fila específica. Ela concederia acesso ao Amazon SNS e a outros serviços da AWS. O Amazon SNS concede uma política padrão a todos os tópicos recém-criados. A política padrão concede acesso ao seu tópico a todos os outros serviços da AWS. Essa política padrão usa uma condição `aws:SourceArn` para garantir que os serviços da AWS acessem seu tópico somente em nome dos recursos da AWS que você possui.

Permitir que notificações de eventos do Simple Storage Service (Amazon S3) publiquem em um tópico

Nesse caso, você deseja configurar a política de um tópico para que outro bucket do Simple Storage Service (Amazon S3) da Conta da AWS possa publicar em seu tópico. Para obter mais informações sobre como publicar notificações do Simple Storage Service (Amazon S3), consulte [Configurar notificações de eventos de bucket](#).

Esse exemplo pressupõe que você criou sua própria política e usou a ação `SetTopicAttributes` para definir o atributo `Policy` do tópico para sua nova política.

A instrução de exemplo a seguir usa a condição `SourceAccount` para garantir que somente a conta de proprietário do Simple Storage Service (Amazon S3) possa acessar o tópico. Neste exemplo, o proprietário do tópico é `111122223333` e o proprietário do Simple Storage Service (Amazon S3) é `444455556666`. O exemplo afirma que qualquer bucket do Amazon S3 de propriedade de `444455556666` tem permissão para publicar no `MyTopic`.

```
{
  "Statement": [{
    "Effect": "Allow",
    "Principal": {
      "Service": "s3.amazonaws.com"
    },
    "Action": "sns:Publish",
    "Resource": "arn:aws:sns:us-east-2:111122223333:MyTopic",
    "Condition": {
      "StringEquals": {
        "AWS:SourceAccount": "444455556666"
      }
    }
  }]
}
```

```
}
```

Ao publicar eventos no Simple Storage Service (Amazon S3), os seguintes serviços oferecem suporte a `aws:SourceAccount`:

- Amazon API Gateway
- Amazon CloudWatch
- DevOpsGuru da Amazon
- Amazon ElastiCache
- Amazon GameLift
- API SMS and Voice do Amazon Pinpoint
- Amazon RDS
- Amazon Redshift
- Amazon S3 Glacier
- Amazon SES
- Amazon Simple Storage Service
- AWS CodeCommit
- AWS Directory Service
- AWS Lambda
- AWS Systems Manager Incident Manager

Permitir que o Amazon SES publique em um tópico pertencente a outra conta

Você pode permitir que outro serviço da AWS publique em um tópico que pertence a outra Conta da AWS. Suponha que você fez login na conta 111122223333, abriu o Amazon SES e criou um e-mail. Para publicar notificações sobre esse e-mail em um tópico do Amazon SNS que a conta 444455556666 possui, crie uma política como a apresentada a seguir. Para fazer isso, você precisa fornecer informações sobre a entidade principal (o outro serviço) e a propriedade de cada recurso. A instrução `Resource` fornece o ARN do tópico, que inclui o ID da conta do respectivo proprietário, 444455556666. A instrução `"aws:SourceOwner": "111122223333"` especifica que sua conta é proprietária do e-mail.

```
{  
  "Version": "2008-10-17",  
  "Id": "__default_policy_ID",
```

```
"Statement": [  
  {  
    "Sid": "__default_statement_ID",  
    "Effect": "Allow",  
    "Principal": {  
      "Service": "ses.amazonaws.com"  
    },  
    "Action": "SNS:Publish",  
    "Resource": "arn:aws:sns:us-east-2:444455556666:MyTopic",  
    "Condition": {  
      "StringEquals": {  
        "aws:SourceOwner": "111122223333"  
      }  
    }  
  }  
]
```

Ao publicar eventos no Amazon SNS, os seguintes serviços comportam `aws:SourceOwner`:

- Amazon API Gateway
- Amazon CloudWatch
- DevOpsGuru da Amazon
- Amazon ElastiCache
- Amazon GameLift
- API SMS and Voice do Amazon Pinpoint
- Amazon RDS
- Amazon Redshift
- Amazon SES
- AWS CodeCommit
- AWS Directory Service
- AWS Lambda
- AWS Systems Manager Incident Manager

## aws:SourceAccount versus aws:SourceOwner

### ⚠ Important

`aws:SourceOwner` foi desativado e novos serviços podem ser integrados ao Amazon SNS somente por meio de `aws:SourceArn` e `aws:SourceAccount`. O Amazon SNS ainda mantém a compatibilidade com versões anteriores dos serviços que no momento são compatíveis com `aws:SourceOwner`.

Cada uma das chaves de condição `aws:SourceAccount` e `aws:SourceOwner` é definida por alguns Serviços da AWS quando eles publicam em um tópico do Amazon SNS. Quando aceitas, o valor será o ID da conta da AWS de 12 dígitos em cujo nome o serviço está publicando dados. Alguns serviços aceitam a primeira e alguns aceitam a segunda.

- Consulte [Permitir que notificações de eventos do Simple Storage Service \(Amazon S3\) publiquem em um tópico](#) para saber como as notificações do Simple Storage Service (Amazon S3) usam `aws:SourceAccount` e uma lista de serviços da AWS que aceita essa condição.
- Consulte [Permitir que o Amazon SES publique em um tópico pertencente a outra conta](#) para saber como o Amazon SES usa `aws:SourceOwner` e uma lista de serviços da AWS que aceitam essa condição.

Permitir que contas em uma organização no AWS Organizations publiquem em um tópico em uma conta diferente

O serviço AWS Organizations ajuda você a gerenciar centralmente o faturamento, controlar o acesso e a segurança e compartilhar recursos entre suas Contas da AWS.

Você pode encontrar o ID da organização no [console do Organizations](#). Para obter mais informações, consulte [Visualização de detalhes de uma organização na conta de gerenciamento](#).

Neste exemplo, qualquer Conta da AWS na organização `myOrgId` pode publicar no `MyTopic` do Amazon SNS na conta `444455556666`. A política verifica o valor do ID da organização usando a chave de condição global `aws:PrincipalOrgID`.

```
{
  "Statement": [
    {
```

```

    "Effect": "Allow",
    "Principal": {
      "AWS": "*"
    },
    "Action": "SNS:Publish",
    "Resource": "arn:aws:sns:us-east-2:444455556666:MyTopic",
    "Condition": {
      "StringEquals": {
        "aws:PrincipalOrgID": "myOrgId"
      }
    }
  }
]
}

```

Permitir que qualquer CloudWatch alarme seja publicado em um tópico em uma conta diferente

Nesse caso, qualquer CloudWatch alarme na conta 111122223333 pode ser publicado em um tópico do Amazon SNS na conta. 444455556666

```

{
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "*"
      },
      "Action": "SNS:Publish",
      "Resource": "arn:aws:sns:us-east-2:444455556666:MyTopic",
      "Condition": {
        "ArnLike": {
          "aws:SourceArn": "arn:aws:cloudwatch:us-
east-2:111122223333:alarm:*"
        }
      }
    }
  ]
}

```

Restringir a publicação em um tópico do Amazon SNS somente de um endpoint da VPC específico

Nesse caso, o tópico na conta 444455556666 está autorizado a publicar somente no endpoint da VPC com o ID vpce-1ab2c34d.

```
{
  "Statement": [{
    "Effect": "Deny",
    "Principal": "*",
    "Action": "SNS:Publish",
    "Resource": "arn:aws:sns:us-east-2:444455556666:MyTopic",
    "Condition": {
      "StringNotEquals": {
        "aws:sourceVpce": "vpce-1ab2c34d"
      }
    }
  }]
}
```

## Como o Amazon Simple Notification Service funciona com o IAM

Antes de usar o IAM para gerenciar o acesso ao Amazon SNS, entenda que recursos do IAM estão disponíveis para uso com o Amazon SNS.

Recursos do IAM que você pode usar com o Amazon Simple Notification Service

Recurso do IAM	Suporte ao Amazon SNS
<a href="#">Políticas baseadas em identidade</a>	Sim
<a href="#">Políticas baseadas em atributos</a>	Sim
<a href="#">Ações de políticas</a>	Sim
<a href="#">Recursos de políticas</a>	Sim
<a href="#">Chaves de condição de política (específicas do serviço)</a>	Sim
<a href="#">ACLs</a>	Não
<a href="#">ABAC (tags em políticas)</a>	Parcial
<a href="#">Credenciais temporárias</a>	Sim
<a href="#">Permissões de entidade principal</a>	Sim

Recurso do IAM	Suporte ao Amazon SNS
<a href="#">Perfis de serviço</a>	Sim
<a href="#">Perfis vinculados ao serviço</a>	Não

Para obter uma visão geral de como o Amazon SNS e outros serviços da AWS funcionam com a maioria dos recursos do IAM, consulte [Serviços da AWS que funcionam com o IAM](#) no Guia do usuário do IAM.

## Ações de políticas para o Amazon SNS

Oferece suporte a ações de políticas Sim

Os administradores podem usar as políticas JSON da AWS para especificar quem tem acesso a quê. Ou seja, qual entidade principal pode executar ações em quais recursos, e em que condições.

O elemento `Action` de uma política JSON descreve as ações que você pode usar para permitir ou negar acesso em uma política. As ações de política geralmente têm o mesmo nome que a operação de API da AWS associada. Existem algumas exceções, como ações somente de permissão, que não têm uma operação de API correspondente. Há também algumas operações que exigem várias ações em uma política. Essas ações adicionais são chamadas de ações dependentes.

Incluem ações em uma política para conceder permissões para executar a operação associada.

Para ver uma lista das ações do Amazon SNS, consulte [Recursos definidos pelo Amazon Simple Notification Service](#) na Referência de autorização do serviço.

As ações de política no Amazon SNS usam o seguinte prefixo antes da ação:

```
sns
```

Para especificar várias ações em uma única instrução, separe-as com vírgulas.

```
"Action": [  
  "sns:action1",  
  "sns:action2"
```

]

Para visualizar exemplos de políticas baseadas em identidade do Amazon SNS, consulte [Exemplos de políticas baseadas em identidade do Amazon Simple Notification Service](#).

## Recursos de políticas para o Amazon SNS

Oferece suporte a recursos de políticas	Sim
---	-----

Os administradores podem usar as políticas JSON da AWS para especificar quem tem acesso a quê. Ou seja, qual entidade principal pode executar ações em quais recursos, e em que condições.

O elemento `Resource` de política JSON especifica o objeto ou os objetos aos quais a ação se aplica. As instruções devem incluir um elemento `Resource` ou um elemento `NotResource`. Como prática recomendada, especifique um recurso usando seu [Nome do recurso da Amazon \(ARN\)](#). Isso pode ser feito para ações que oferecem suporte a um tipo de recurso específico, conhecido como permissões em nível de recurso.

Para ações que não oferecem suporte a permissões em nível de recurso, como operações de listagem, use um caractere curinga (\*) para indicar que a instrução se aplica a todos os recursos.

```
"Resource": "*"

```

Para ver uma lista de tipos de recurso do Amazon SNS e seus ARNs, consulte [Ações definidas pelo Amazon Simple Notification Service](#) na Referência de autorização do serviço. Para saber com quais ações você pode especificar o ARN de cada recurso, consulte [Recursos definidos pelo Amazon Simple Notification Service](#).

Para visualizar exemplos de políticas baseadas em identidade do Amazon SNS, consulte [Exemplos de políticas baseadas em identidade do Amazon Simple Notification Service](#).

## Chaves de condição de política para o Amazon SNS

Compatível com chaves de condição de política específicas do serviço	Sim
--	-----

Os administradores podem usar as políticas JSON da AWS para especificar quem tem acesso a quê. Ou seja, qual entidade principal pode executar ações em quais recursos e em que condições.

O elemento `Condition` (ou bloco de `Condition`) permite que você especifique condições nas quais uma instrução está em vigor. O elemento `Condition` é opcional. É possível criar expressões condicionais que usam [agentes de condição](#), como “igual a” ou “menor que”, para fazer a condição da política corresponder aos valores na solicitação.

Se você especificar vários elementos `Condition` em uma instrução ou várias chaves em um único elemento `Condition`, a AWS os avaliará usando uma operação lógica AND. Se você especificar vários valores para uma única chave de condição, a AWS avaliará a condição usando uma operação lógica OR. Todas as condições devem ser atendidas para que as permissões da instrução sejam concedidas.

Você também pode usar variáveis de espaço reservado ao especificar as condições. Por exemplo, é possível conceder a um usuário do IAM permissão para acessar um recurso somente se ele estiver marcado com seu nome de usuário do IAM. Para obter mais informações, consulte [Elementos de política do IAM: variáveis e tags](#) no Guia do usuário do IAM.

A AWS oferece suporte a chaves de condição globais e chaves de condição específicas do serviço. Para ver todas as AWS chaves de condição globais da , consulte [AWSChaves de contexto de condição globais da](#) no Guia do usuário do IAM.

Para ver uma lista de chaves de condição do Amazon SNS, consulte [Chaves de condição do Amazon Simple Notification Service](#) na Referência de autorização do serviço. Para saber com quais ações e recursos você pode usar a chave de condição, consulte [Recursos definidos pelo Amazon Simple Notification Service](#).

Para visualizar exemplos de políticas baseadas em identidade do Amazon SNS, consulte [Exemplos de políticas baseadas em identidade do Amazon Simple Notification Service](#).

## ACLs no Amazon SNS

Oferece suporte a ACLs

Não

As listas de controle de acesso (ACLs) controlam quais principais (membros, usuários ou funções da conta) têm permissões para acessar um recurso. As ACLs são semelhantes às políticas baseadas em recursos, embora não usem o formato de documento de política JSON.

## ABAC com o Amazon SNS

Oferece suporte a ABAC (tags em políticas)      Parcial

O controle de acesso baseado em atributo (ABAC) é uma estratégia de autorização que define permissões com base em atributos. Na AWS, esses atributos são chamados de tags. É possível anexar tags a entidades do IAM (usuários ou funções) e a muitos recursos da AWS. A marcação de entidades e recursos é a primeira etapa do ABAC. Em seguida, você cria políticas de ABAC para permitir operações quando a tag da entidade principal corresponder à tag do recurso que ela está tentando acessar.

O ABAC é útil em ambientes que estão crescendo rapidamente e ajuda em situações em que o gerenciamento de políticas se torna um problema.

Para controlar o acesso baseado em tags, forneça informações sobre as tags no [elemento de condição](#) de uma política usando as `aws:ResourceTag/key-name`, `aws:RequestTag/key-name` ou `aws:TagKeys` chaves de condição.

Se um serviço oferecer suporte às três chaves de condição para cada tipo de recurso, o valor será Sim para o serviço. Se um serviço oferecer suporte às três chaves de condição somente para alguns tipos de recursos, o valor será Parcial.

Para obter mais informações sobre o ABAC, consulte [O que é ABAC?](#) no Guia do usuário do IAM. Para visualizar um tutorial com etapas para configurar o ABAC, consulte [Usar controle de acesso baseado em atributos \(ABAC\)](#) no Guia do usuário do IAM.

## Usar credenciais temporárias com o Amazon SNS

Oferece suporte a credenciais temporárias      Sim

Alguns Serviços da AWS não funcionam quando você faz login usando credenciais temporárias. Para obter informações adicionais, incluindo quais Serviços da AWS funcionam com credenciais temporárias, consulte [Serviços da AWS que funcionam com o IAM](#) no Guia do usuário do IAM.

Você está usando credenciais temporárias se faz login no AWS Management Console usando qualquer método, exceto um nome de usuário e uma senha. Por exemplo, quando você

acessa a AWS usando o link de autenticação única (SSO) da sua empresa, esse processo cria automaticamente credenciais temporárias. Você também cria automaticamente credenciais temporárias quando faz login no console como usuário e, em seguida, alterna funções. Para obter mais informações sobre como alternar funções, consulte [Alternar para uma função \(console\)](#) no Guia do usuário do IAM.

Você pode criar credenciais temporárias manualmente usando a AWS CLI ou a API da AWS. Em seguida, você pode usar essas credenciais temporárias para acessar a AWS. A AWS recomenda que você gere credenciais temporárias dinamicamente em vez de usar chaves de acesso de longo prazo. Para obter mais informações, consulte [Credenciais de segurança temporárias no IAM](#).

## Permissões de entidades principais entre serviços para o Amazon SNS

Suporte para o recurso Encaminhamento de sessões de acesso (FAS)	Sim
--	-----

Quando você usa um usuário ou um perfil do IAM para executar ações na AWS, você é considerado uma entidade principal. Ao usar alguns serviços, você pode executar uma ação que inicia outra ação em um serviço diferente. O recurso FAS utiliza as permissões da entidade principal que chama um AWS service (Serviço da AWS), combinadas às permissões do AWS service (Serviço da AWS) solicitante, para realizar solicitações para serviços downstream. As solicitações de FAS só são feitas quando um serviço recebe uma solicitação que exige interações com outros Serviços da AWS ou com recursos para serem concluídas. Nesse caso, você precisa ter permissões para executar ambas as ações. Para obter detalhes da política ao fazer solicitações de FAS, consulte [Encaminhar sessões de acesso](#).

## Perfis de serviço para o Amazon SNS

Oferece suporte a perfis de serviço	Sim
-------------------------------------	-----

O perfil de serviço é um [perfil do IAM](#) que um serviço assume para realizar ações em seu nome. Um administrador do IAM pode criar, modificar e excluir um perfil de serviço do IAM. Para obter mais informações, consulte [Criar um perfil para delegar permissões a um AWS service \(Serviço da AWS\)](#) no Guia do usuário do IAM.

**⚠ Warning**

A alteração das permissões de um perfil de serviço pode interromper a funcionalidade do Amazon SNS. Edite perfis de serviço somente quando o Amazon SNS fornecer orientação para isso.

## Perfis vinculados ao serviço para o Amazon SNS

Oferece suporte a perfis vinculados ao serviço      Não

Um perfil vinculado ao serviço é um tipo de perfil de serviço vinculado a um AWS service (Serviço da AWS). O serviço pode assumir a função de executar uma ação em seu nome. Os perfis vinculados ao serviço aparecem em sua Conta da AWS e são de propriedade do serviço. Um administrador do IAM pode visualizar, mas não pode editar as permissões para perfis vinculados ao serviço.

Para obter detalhes sobre como criar ou gerenciar funções vinculadas a serviços, consulte [Serviços do AWS que funcionam com o IAM](#). Encontre um serviço na tabela que inclua um Yes na coluna Função vinculada ao serviço. Escolha o link Sim para visualizar a documentação do perfil vinculado a esse serviço.

## Exemplos de políticas baseadas em identidade do Amazon Simple Notification Service

Por padrão, os usuários e perfis não têm permissão para criar ou modificar recursos do Amazon SNS. Eles também não podem executar tarefas usando o AWS Management Console, a AWS Command Line Interface (AWS CLI) ou a API AWS. Para conceder aos usuários permissões para executar ações nos recursos de que eles precisam, um administrador do IAM pode criar políticas do IAM. O administrador pode então adicionar as políticas do IAM aos perfis, e os usuários podem assumir os perfis.

Para saber como criar uma política baseada em identidade do IAM usando esses exemplos de documento de política JSON, consulte [Criação de políticas do IAM](#) no Guia do Usuário do IAM.

Para obter detalhes sobre ações e tipos de recurso definidos pelo Amazon SNS, incluindo o formato dos ARNs para cada um dos tipos de recurso, consulte [Ações, recursos e chaves de condição do Amazon Simple Notification Service](#) na Referência de autorização do serviço.

## Tópicos

- [Práticas recomendadas de políticas](#)
- [Usar o console do Amazon SNS](#)
- [Outros tipos de política](#)
- [Vários tipos de política](#)
- [Permitir que os usuários visualizem suas próprias permissões](#)

## Práticas recomendadas de políticas

As políticas baseadas em identidade determinam se alguém pode criar, acessar ou excluir recursos do Amazon SNS em sua conta. Essas ações podem incorrer em custos para a Conta da AWS. Ao criar ou editar políticas baseadas em identidade, siga estas diretrizes e recomendações:

- Comece com as políticas gerenciadas pela AWS e avance para as permissões de privilégio mínimo: para começar a conceder permissões a seus usuários e workloads, use as políticas gerenciadas pela AWS que concedem permissões para muitos casos de uso comuns. Eles estão disponíveis na sua Conta da AWS. Recomendamos que você reduza ainda mais as permissões definindo políticas gerenciadas pelo cliente da AWS específicas para seus casos de uso. Para obter mais informações, consulte [Políticas gerenciadas pela AWS](#) ou [Políticas gerenciadas pela AWS para funções de trabalho](#) no Guia do usuário do IAM.
- Aplique permissões de privilégio mínimo: ao definir permissões com as políticas do IAM, conceda apenas as permissões necessárias para executar uma tarefa. Você faz isso definindo as ações que podem ser executadas em recursos específicos sob condições específicas, também conhecidas como permissões de privilégio mínimo. Para obter mais informações sobre como usar o IAM para aplicar permissões, consulte [Políticas e permissões no IAM](#) no Guia do usuário do IAM.
- Use condições nas políticas do IAM para restringir ainda mais o acesso: você pode adicionar uma condição às políticas para limitar o acesso a ações e recursos. Por exemplo, você pode escrever uma condição de política para especificar que todas as solicitações devem ser enviadas usando SSL. Você também pode usar condições para conceder acesso a ações de serviço, se elas forem usadas por meio de um AWS service (Serviço da AWS) específico, como o AWS CloudFormation. Para obter mais informações, consulte [Elementos de política JSON do IAM: Condition](#) no Manual do usuário do IAM.
- Use o IAM Access Analyzer para validar suas políticas do IAM a fim de garantir permissões seguras e funcionais: o IAM Access Analyzer valida as políticas novas e existentes para que elas sigam a linguagem de política do IAM (JSON) e as práticas recomendadas do IAM. O IAM Access

Analyzer oferece mais de 100 verificações de política e recomendações acionáveis para ajudar você a criar políticas seguras e funcionais. Para obter mais informações, consulte [Validação de políticas do IAM Access Analyzer](#) no Guia do usuário do IAM.

- Exigir autenticação multifator (MFA): se houver um cenário que exija usuários do IAM ou um usuário raiz em sua Conta da AWS, ative a MFA para obter segurança adicional. Para exigir a MFA quando as operações de API forem chamadas, adicione condições de MFA às suas políticas. Para obter mais informações, consulte [Configuração de acesso à API protegido por MFA](#) no Guia do usuário do IAM.

Para obter mais informações sobre as práticas recomendadas do IAM, consulte [Práticas recomendadas de segurança no IAM](#) no Guia do usuário do IAM.

## Usar o console do Amazon SNS

Para acessar o console do Amazon Simple Notification Service, você deve ter um conjunto mínimo de permissões. Essas permissões devem permitir que você liste e visualize detalhes sobre os recursos do Amazon SNS em sua Conta da AWS. Se você criar uma política baseada em identidade que seja mais restritiva do que as permissões mínimas necessárias, o console não funcionará como pretendido para entidades (usuários ou perfis) com essa política.

Não é necessário conceder permissões mínimas do console para usuários que fazem chamadas somente à AWS CLI ou à AWS API. Em vez disso, permita o acesso somente a ações que correspondam a operação de API que estiverem tentando executar.

Para garantir que usuários e perfis ainda possam usar o console do Amazon SNS, anexe também a política *ConsoleAccess* ou *ReadOnly* gerenciada pela AWS do Amazon SNS às entidades. Para obter mais informações, consulte [Adicionando Permissões a um Usuário](#) no Guia do Usuário do IAM.

## Outros tipos de política

A AWS aceita tipos de política menos comuns. Esses tipos de política podem definir o máximo de permissões concedidas a você pelos tipos de política mais comuns.

- Limites de permissões: um limite de permissões é um recurso avançado no qual você define o máximo de permissões que uma política baseada em identidade pode conceder a uma entidade do IAM (usuário ou perfil do IAM). É possível definir um limite de permissões para uma entidade. As permissões resultantes são a interseção das políticas baseadas em identidade de uma entidade e dos seus limites de permissões. As políticas baseadas em recurso que especificam

o usuário ou a função no campo `Principal` não são limitadas pelo limite de permissões. Uma negação explícita em qualquer uma dessas políticas substitui a permissão. Para obter mais informações sobre limites de permissões, consulte [Limites de permissões para identidades do IAM](#) no Guia do usuário do IAM.

- **Políticas de controle de serviço (SCPs):** SCPs são políticas JSON que especificam as permissões máximas para uma organização ou unidade organizacional (UO) no AWS Organizations. O AWS Organizations é um serviço para agrupar e gerenciar centralmente várias Contas da AWS pertencentes à sua empresa. Se você habilitar todos os recursos em uma organização, poderá aplicar políticas de controle de serviço (SCPs) a qualquer uma ou a todas as contas. Uma SCP limita as permissões para entidades em contas-membro, inclusive para cada usuário raiz da Conta da AWS. Para obter mais informações sobre o Organizations e SCPs, consulte [Como os SCPs funcionam](#) no Guia do usuário do AWS Organizations.
- **Políticas de sessão:** são políticas avançadas que você transmite como um parâmetro quando cria de forma programática uma sessão temporária para uma função ou um usuário federado. As permissões da sessão resultante são a interseção das políticas baseadas em identidade do usuário ou da função e das políticas de sessão. As permissões também podem ser provenientes de uma política baseada em recurso. Uma negação explícita em qualquer uma dessas políticas substitui a permissão. Para obter mais informações, consulte [Políticas de sessão](#) no Guia do usuário do IAM.

## Vários tipos de política

Quando vários tipos de política são aplicáveis a uma solicitação, é mais complicado compreender as permissões resultantes. Para saber como a AWS determina se deve permitir uma solicitação quando há vários tipos de política envolvidos, consulte [Lógica da avaliação](#) de políticas no Guia do usuário do IAM.

## Permitir que os usuários visualizem suas próprias permissões

Este exemplo mostra como você pode criar uma política que permite que os usuários do IAM visualizem as políticas gerenciadas e em linha anexadas a sua identidade de usuário. Essa política inclui permissões para concluir essa ação no console ou de forma programática usando a AWS CLI ou a AWS API.

```
{
  "Version": "2012-10-17",
  "Statement": [
```

```
{
  "Sid": "ViewOwnUserInfo",
  "Effect": "Allow",
  "Action": [
    "iam:GetUserPolicy",
    "iam:ListGroupsWithUser",
    "iam:ListAttachedUserPolicies",
    "iam:ListUserPolicies",
    "iam:GetUser"
  ],
  "Resource": ["arn:aws:iam::*:user/${aws:username}"]
},
{
  "Sid": "NavigateInConsole",
  "Effect": "Allow",
  "Action": [
    "iam:GetGroupPolicy",
    "iam:GetPolicyVersion",
    "iam:GetPolicy",
    "iam:ListAttachedGroupPolicies",
    "iam:ListGroupPolicies",
    "iam:ListPolicyVersions",
    "iam:ListPolicies",
    "iam:ListUsers"
  ],
  "Resource": "*"
}
]
```

## Políticas baseadas em identidade do Amazon SNS

É compatível com políticas baseadas em identidade	Sim
---	-----

As políticas baseadas em identidade são documentos de políticas de permissões JSON que você pode anexar a uma identidade, como usuário, grupo de usuários ou perfil do IAM. Essas políticas controlam quais ações os usuários e funções podem realizar, em quais recursos e em que condições. Para saber como criar uma política baseada em identidade, consulte [Criar políticas do IAM](#) no Guia do usuário do IAM.

Com as políticas baseadas em identidade do IAM, é possível especificar ações ou recursos permitidos ou negados, bem como as condições sob as quais as ações são permitidas ou negadas. Você não pode especificar a entidade principal em uma política baseada em identidade porque ela se aplica ao usuário ou função à qual ela está anexado. Para saber mais sobre todos os elementos que podem ser usados em uma política JSON, consulte [Referência de elementos da política JSON do IAM](#) no Guia do Usuário do IAM.

## Exemplos de políticas baseadas em identidade para o Amazon SNS

Para visualizar exemplos de políticas baseadas em identidade do Amazon SNS, consulte [Exemplos de políticas baseadas em identidade do Amazon Simple Notification Service](#).

## Políticas baseadas em recursos no Amazon SNS

Oferece suporte a políticas baseadas em atributos	Sim
---	-----

Políticas baseadas em recurso são documentos de políticas JSON que você anexa a um recurso. São exemplos de políticas baseadas em recursos as políticas de confiança de perfil do IAM e as políticas de bucket do Amazon S3. Em serviços compatíveis com políticas baseadas em recursos, os administradores de serviço podem usá-las para controlar o acesso a um recurso específico. Para o recurso ao qual a política está anexada, a política define quais ações um principal especificado pode executar nesse recurso e em que condições. Você deve [especificar um principal](#) em uma política baseada em recursos. As entidades principais podem incluir contas, usuários, funções, usuários federados ou Serviços da AWS.

Para permitir o acesso entre contas, você pode especificar uma conta inteira ou as entidades do IAM em outra conta como a entidade principal em uma política baseada em recurso. Adicionar um principal entre contas à política baseada em recurso é apenas metade da tarefa de estabelecimento da relação de confiança. Quando a entidade principal e o recurso estão em diferentes Contas da AWS, um administrador do IAM da conta confiável também deve conceder à entidade principal (usuário ou função) permissão para acessar o recurso. Eles concedem permissão ao anexar uma política baseada em identidade para a entidade. No entanto, se uma política baseada em recurso conceder acesso a um principal na mesma conta, nenhuma política baseada em identidade adicional será necessária. Para obter mais informações, consulte [Como os perfis do IAM diferem de políticas baseadas em recursos](#) no Guia do usuário do IAM.

# Usar políticas baseadas em identidade com o Amazon SNS

## Tópicos

- [Políticas do IAM e do Amazon SNS juntas](#)
- [Formato do ARN de recurso do Amazon SNS](#)
- [Ações da API do Amazon SNS](#)
- [Chaves de política do Amazon SNS](#)
- [Exemplos de políticas do Amazon SNS](#)

O Amazon Simple Notification Service integra-se ao AWS Identity and Access Management (IAM) para que você possa especificar quais ações do Amazon SNS um usuário em sua Conta da AWS pode executar com recursos do Amazon SNS. Você pode especificar um tópico específico na política. Por exemplo, você poderá usar variáveis ao criar uma política do IAM que conceda a determinados usuários em sua organização permissão para usar a ação Publish com tópicos específicos em sua Conta da AWS. Para obter mais informações, consulte [Variáveis de políticas](#) no guia Como usar o IAM.

### Important

O uso do Amazon SNS com o IAM não altera a forma como você usa o Amazon SNS. Não há alterações nas ações do Amazon SNS nem novas ações do Amazon SNS relacionadas a controle de acesso e usuários.

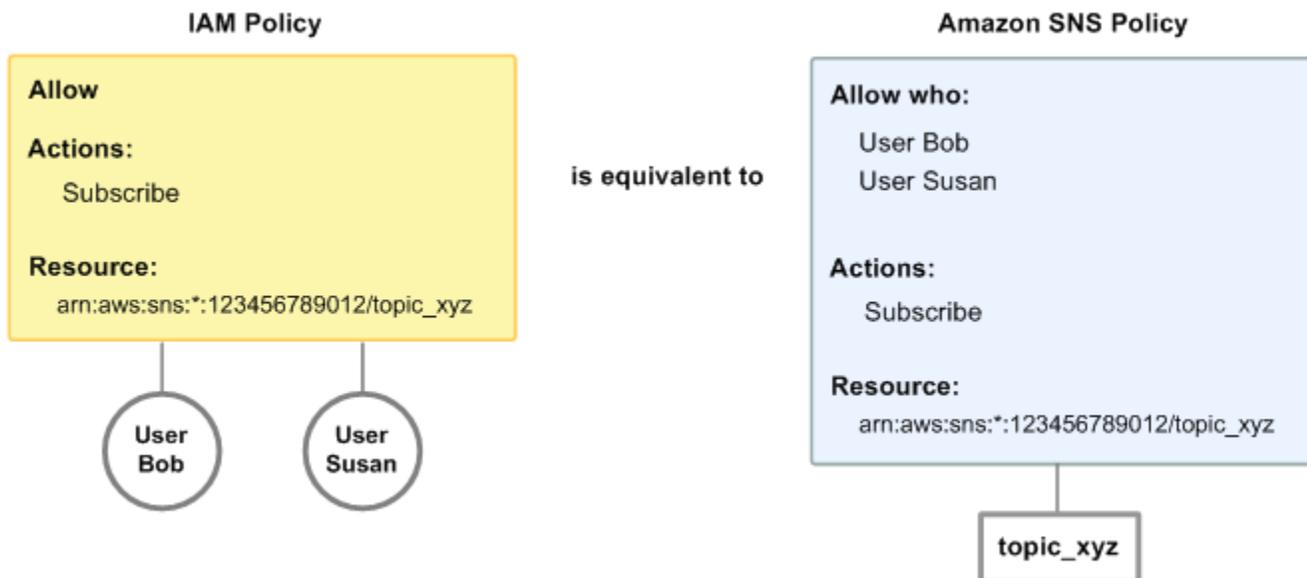
Para obter exemplos de políticas que abrangem ações e recursos do Amazon SNS, consulte [Exemplos de políticas do Amazon SNS](#).

## Políticas do IAM e do Amazon SNS juntas

Você usa uma política do IAM para restringir o acesso dos usuários a ações e tópicos do Amazon SNS. Uma política do IAM pode restringir o acesso apenas a usuários em sua conta da AWS, não a outras Contas da AWS.

Você usa uma política do Amazon SNS com determinado tópico para restringir quem pode trabalhar com esse tópico (por exemplo, quem pode publicar mensagens nele, quem pode assiná-lo etc.). As políticas do Amazon SNS podem conceder acesso a outras Contas da AWS ou a usuários dentro de sua Conta da AWS.

Para conceder permissão para seus tópicos do Amazon SNS aos usuários, você pode usar políticas do IAM, políticas do Amazon SNS ou ambas. Na maior parte das vezes, você obtém os mesmos resultados com qualquer uma delas. Por exemplo, o diagrama a seguir mostra uma política do IAM e uma política do Amazon SNS que são equivalentes. A política do IAM permite a ação `Subscribe` do Amazon SNS para o tópico denominado `topic_xyz` em sua conta da AWS. A política do IAM é associada aos usuários Pedro e Ana (o que significa que Pedro e Ana têm as permissões declaradas na política). A política do Amazon SNS também concede a Pedro e Ana permissão para acessar `Subscribe` para `topic_xyz`.



### Note

O exemplo anterior mostra políticas simples, sem condições. Você pode especificar determinada condição na política e obter o mesmo resultado.

Há uma diferença entre as políticas do AWS IAM e do Amazon SNS: o sistema de políticas do Amazon SNS permite conceder permissões a outras Contas da AWS, enquanto a política do IAM não permite.

Você decide como usar ambos os sistemas juntos para gerenciar suas permissões, de acordo com suas necessidades. Os exemplos a seguir mostram como os dois sistemas de política funcionam em conjunto.

## Example 1

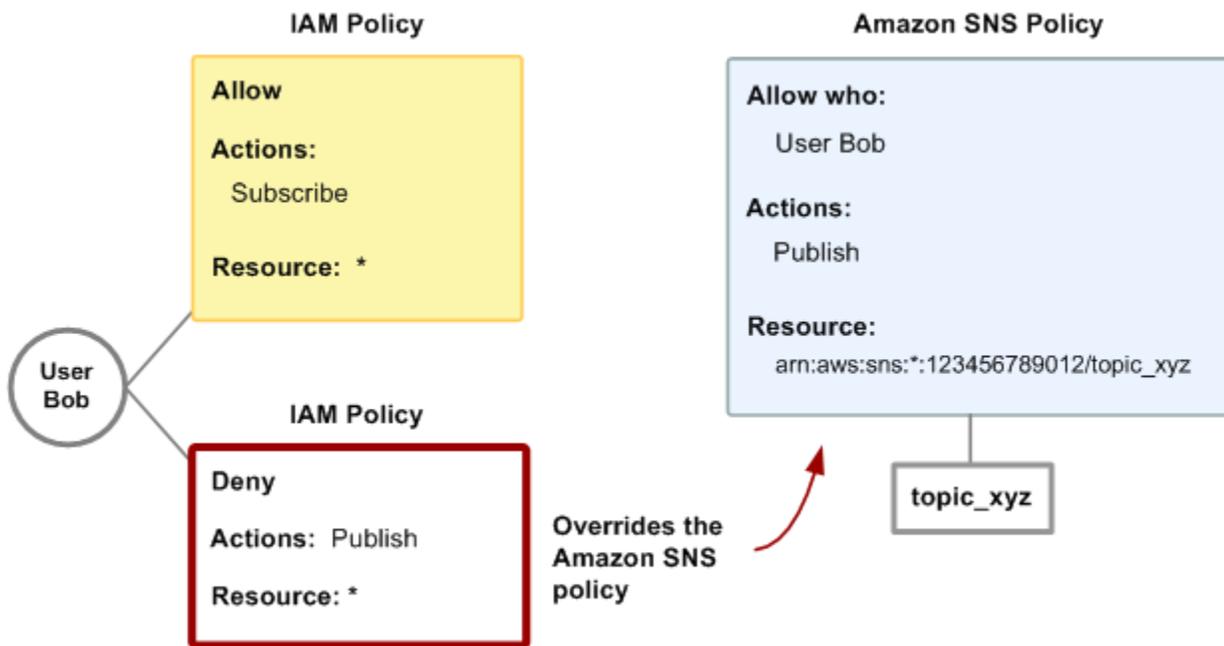
Neste exemplo, uma política do IAM e uma política do Amazon SNS se aplicam a Pedro. A política do IAM concede a ele permissão para `Subscribe` em qualquer um dos tópicos da Conta da AWS, enquanto a política do Amazon SNS concede a ele permissão para usar `Publish` em um tópico específico (`topic_xyz`). O seguinte diagrama ilustra o conceito.



Se Pedro enviasse uma solicitação para assinar um tópico na conta da AWS, a política do IAM permitiria a ação. Se Pedro enviasse uma solicitação para publicar uma mensagem no `topic_xyz`, a política do Amazon SNS permitiria a ação.

## Example 2

Neste exemplo, nós nos baseamos no exemplo 1 (onde há duas políticas que se aplicam a Pedro). Digamos que Pedro publique mensagens no tópico `topic_xyz` que não deveria ter publicado. Por isso, você deseja remover totalmente a capacidade dele de publicar nos tópicos. O mais fácil a fazer é adicionar uma política do IAM que negue acesso a ele à ação `Publish` em todos os tópicos. Essa terceira política se sobrepõe à política do Amazon SNS que originalmente deu a ele permissão para publicar no `topic_xyz`, pois uma negação explícita sempre se sobrepõe a uma permissão (para obter mais informações sobre a lógica de avaliação de políticas, consulte [Lógica de avaliação](#)). O seguinte diagrama ilustra o conceito.



Para obter exemplos de políticas que abrangem ações e recursos do Amazon SNS, consulte [Exemplos de políticas do Amazon SNS](#). Para obter mais informações sobre como escrever políticas do Amazon SNS, consulte a [documentação técnica do Amazon SNS](#).

## Formato do ARN de recurso do Amazon SNS

Para o Amazon SNS, os tópicos são o único tipo de recurso que você pode especificar em uma política. Veja a seguir o formato de nome do recurso da Amazon (ARN) para tópicos.

```
arn:aws:sns:region:account_ID:topic_name
```

Para obter mais informações sobre ARNs, consulte [ARNs](#) no guia do usuário do usuário do IAM.

### Example

A seguir está um ARN para um tópico nomeado MyTopic na região us-east-2, pertencente a 123456789012. Conta da AWS

```
arn:aws:sns:us-east-2:123456789012:MyTopic
```

## Example

Se você tivesse um tópico nomeado MyTopic em cada uma das diferentes regiões suportadas pelo Amazon SNS, você poderia especificar os tópicos com o seguinte ARN.

```
arn:aws:sns:*:123456789012:MyTopic
```

Você pode usar os caracteres curinga \* e ? no nome do tópico. Por exemplo, o seguinte poderia fazer referência a todos os tópicos criados por Pedro nos quais ele inseriu o prefixo bob\_.

```
arn:aws:sns:*:123456789012:bob_*
```

Por uma questão de conveniência para você, ao criar um tópico, o Amazon SNS retorna o ARN do tópico como resposta.

## Ações da API do Amazon SNS

Em uma política do IAM, você pode especificar quaisquer ações que o Amazon SNS oferecer. No entanto, as ações `ConfirmSubscription` e `Unsubscribe` não exigem autenticação, o que significa que, mesmo que você especifique essas ações em uma política, o IAM não restringirá o acesso dos usuários a essas ações.

Cada ação que você especificar em uma política deve ser prefixada com a string `sns:` em letras minúsculas. Por exemplo, para especificar todas as ações do Amazon SNS, você usaria `sns:*`. Para obter uma lista de ações, consulte [Amazon Simple Notification Service](#).

## Chaves de política do Amazon SNS

O Amazon SNS implementa as seguintes chaves de política em toda a AWS, além de algumas chaves específicas do serviço.

Para obter uma lista de chaves de condição compatíveis com cada serviço da AWS service (Serviço da AWS), consulte [Ações, recursos e chaves de condição de Serviços da AWS](#) no Guia do usuário do IAM. Para obter uma lista de chaves de condição que podem ser usadas em vários Serviços da AWS, consulte [Chaves de contexto de condição globais da AWS](#) no Guia do usuário do IAM.

O Amazon SNS usa as chaves específicas de serviço a seguir. Use essas chaves em políticas que restringem o acesso a solicitações de `Subscribe`.

- `sns:endpoint`: o URL, o endereço de e-mail ou o ARN de uma solicitação `Subscribe` ou de uma assinatura confirmada anteriormente. Use com condições de string (consulte [Exemplos](#)

[de políticas do Amazon SNS](#)) para restringir o acesso a endpoints específicos (por exemplo, \*@yourcompany.com).

- sns:protocol: o valor protocol de uma solicitação Subscribe ou de uma assinatura confirmada anteriormente. Use com condições de string (consulte [Exemplos de políticas do Amazon SNS](#)) para restringir a publicação a protocolos de entrega específicos (por exemplo, https).

## Exemplos de políticas do Amazon SNS

Esta seção mostra várias políticas simples para controlar o acesso de usuários ao Amazon SNS.

### Note

No futuro, o Amazon SNS pode adicionar novas ações que devem ser logicamente incluídas em uma das seguintes políticas, com base nos objetivos indicados da política.

### Example 1: Permitir que um grupo crie e gerencie tópicos

Neste exemplo, criamos uma política que concede acesso a CreateTopic, ListTopics, SetTopicAttributes e DeleteTopic.

```
{
  "Statement": [{
    "Effect": "Allow",
    "Action": ["sns:CreateTopic", "sns:ListTopics", "sns:SetTopicAttributes",
"sns>DeleteTopic"],
    "Resource": "*"
  }]
}
```

### Example 2: Permitir que o grupo de TI publique mensagens em determinado tópico

Neste exemplo, criamos um grupo de TI e atribuímos uma política que concede acesso a Publish no tópico de interesse específico.

```
{
  "Statement": [{
    "Effect": "Allow",
    "Action": "sns:Publish",
```

```
"Resource": "arn:aws:sns:*:123456789012:MyTopic"
}]
}
```

### Example 3: Conceder aos usuários na Conta da AWS a capacidade de assinar tópicos

Neste exemplo, criamos uma política que concede acesso à ação `Subscribe`, com as condições de correspondência de string para as chaves de política `sns:Protocol` e `sns:Endpoint`.

```
{
  "Statement": [{
    "Effect": "Allow",
    "Action": ["sns:Subscribe"],
    "Resource": "*",
    "Condition": {
      "StringLike": {
        "SNS:Endpoint": "*@example.com"
      },
      "StringEquals": {
        "sns:Protocol": "email"
      }
    }
  }]
}
```

### Example 4: Permitir que um parceiro publique mensagens em determinado tópico

Você pode usar uma política do Amazon SNS ou uma política do IAM para permitir que um parceiro publique em um tópico específico. Se seu parceiro tiver uma Conta da AWS, pode ser mais fácil usar uma política do Amazon SNS. No entanto, qualquer pessoa na empresa do parceiro que tiver acesso às credenciais de segurança da AWS poderá publicar mensagens no tópico. Este exemplo pressupõe que você deseje limitar o acesso a determinada pessoa (ou aplicação). Para fazer isso, é preciso tratar o parceiro como um usuário dentro de sua própria empresa e usar uma política do IAM em vez de uma política do Amazon SNS.

Neste exemplo, criamos um grupo chamado `WidgetCo` que representa a empresa parceira; criamos um usuário para a pessoa específica (ou aplicativo) na empresa parceira que precisa de acesso; e então colocamos o usuário no grupo.

Em seguida, anexamos uma política que concede ao grupo `Publish` acesso ao tópico específico nomeado `WidgetPartnerTopic`.

Também queremos evitar que o WidgetCo grupo faça qualquer outra coisa com tópicos, então adicionamos uma declaração que nega permissão para qualquer ação do Amazon SNS, Publish exceto em qualquer outro tópico que não seja. WidgetPartnerTopic Isso será necessário somente se houver uma política ampla em outro lugar no sistema que conceda aos usuários amplo acesso ao Amazon SNS.

```
{
  "Statement": [{
    "Effect": "Allow",
    "Action": "sns:Publish",
    "Resource": "arn:aws:sns:*:123456789012:WidgetPartnerTopic"
  },
  {
    "Effect": "Deny",
    "NotAction": "sns:Publish",
    "NotResource": "arn:aws:sns:*:123456789012:WidgetPartnerTopic"
  }
  ]
}
```

## Usar credenciais de segurança temporárias com o Amazon SNS

Além de criar usuários do IAM com suas próprias credenciais de segurança, o IAM também permite conceder credenciais de segurança temporárias a qualquer usuário, possibilitando que ele acesse os recursos e serviços da AWS. Você pode gerenciar os usuários que têm Contas da AWS. Eles são usuários do IAM. Você também pode gerenciar usuários para seu sistema que não tenham Contas da AWS. Eles são chamados de usuários federados. Além disso, os “usuários” também podem ser aplicações que você cria para acessar seus recursos da AWS.

Use as credenciais de segurança temporárias para fazer solicitações ao Amazon SNS. As bibliotecas de API calculam o valor de assinatura necessário usando essas credenciais para autenticar sua solicitação. Se você enviar solicitações usando credenciais vencidas, o Amazon SNS negará a solicitação.

Para obter mais informações sobre o suporte do IAM para credenciais de segurança temporárias, acesse [Conceder acesso temporário aos recursos da AWS](#) em Usar o IAM.

## Example Usar credenciais de segurança temporárias para autenticar uma solicitação do Amazon SNS

O exemplo a seguir demonstra como obter credenciais de segurança temporárias para autenticar uma solicitação do Amazon SNS.

```
http://sns.us-east-2.amazonaws.com/  
?Name=My-Topic  
&Action=CreateTopic  
&Signature=gfzIF53exFVdpSNb8AiwN3Lv%2FNYXh6S%2Br3yySK70oX4%3D  
&SignatureVersion=2  
&SignatureMethod=HmacSHA256  
&Timestamp=2010-03-31T12%3A00%3A00.000Z  
&SecurityToken=SecurityTokenValue  
&AWSAccessKeyId=Access Key ID provided by AWS Security Token Service
```

## Permissões da API do Amazon SNS: referência de ações e recursos

A lista a seguir fornece informações específicas para a implementação de controle de acesso do Amazon SNS:

- Cada política deve abranger apenas um tópico (ao gravar uma política, não inclua declarações que abrangem diferentes tópicos).
- Cada política deve ter uma política exclusiva Id.
- Cada declaração em uma política deve ter uma declaração exclusiva sid.

### Cotas de política

A tabela a seguir lista as cotas máximas para uma declaração de política.

Nome	Cota máxima
Bytes	30 kb
Declarações	100
Entidades principais	1 a 200 (0 é inválido.)

Nome	Cota máxima
Recurso	1 (0 é inválido. O valor deve corresponder ao ARN do tópico da política.)

## Ações de política válidas do Amazon SNS

O Amazon SNS é compatível com as ações mostradas na tabela a seguir.

Ação	Descrição
sns: AddPermission	Concede permissão para adicionar permissões à política do tópico.
sns: DeleteTopic	Concede permissão para excluir um tópico.
sns: GetDataProtectionPolicy	Concede permissão para recuperar a política de proteção de dados de um tópico.
sns: GetTopicAttributes	Concede permissão para receber todos os atributos de tópico.
sns: ListSubscriptionsByTopic	Concede permissão para recuperar todas as assinaturas de determinado tópico.
sns: ListTagsForResource	Concede permissão para relacionar todas as etiquetas adicionadas a um tópico específico.
sns: Publish	Concede permissão para publicar individualmente ou em lote em um tópico ou endpoint. Para obter mais informações, consulte <a href="#">Publish</a> e <a href="#">PublishBatch</a> na Referência da API do Amazon Simple Notification Service.
sns: PutDataProtectionPolicy	Concede permissão para definir a política de proteção de dados de um tópico.
sns: RemovePermission	Concede permissão para remover as permissões na política de tópico.
sns: SetTopicAttributes	Concede permissão para definir os atributos de um tópico.

Ação	Descrição
sns:Subscribe	Concede permissão para assinar um tópico.

## Chaves específicas do serviço

O Amazon SNS usa as chaves específicas de serviço a seguir. Você pode usar essas chaves nas políticas que restringem o acesso a solicitações `Subscribe`.

- `sns:endpoint`: o URL, o endereço de e-mail ou o ARN de uma solicitação `Subscribe` ou de uma assinatura confirmada anteriormente. Use com condições de string (consulte [Exemplos de políticas do Amazon SNS](#)) para restringir o acesso a endpoints específicos (por exemplo, `*@example.com`).
- `sns:protocol`: o valor `protocol` de uma solicitação `Subscribe` ou de uma assinatura confirmada anteriormente. Use com condições de string (consulte [Exemplos de políticas do Amazon SNS](#)) para restringir a publicação a protocolos de entrega específicos (por exemplo, `https`).

### Important

Quando você usar uma política para controlar o acesso por `sns:Endpoint`, lembre-se de que os problemas de DNS podem afetar a resolução de nomes de endpoints no futuro.

## Resolução de problemas de identidade e acesso do Amazon Simple Notification Service

Use as informações a seguir para ajudar a diagnosticar e corrigir problemas comuns que você pode encontrar ao trabalhar com o Amazon SNS e o IAM.

### Tópicos

- [Não tenho autorização para executar uma ação no Amazon SNS](#)
- [Não estou autorizado a realizar iam: PassRole](#)
- [Quero permitir que as pessoas fora da minha Conta da AWS acessem meus recursos do Amazon SNS](#)

## Não tenho autorização para executar uma ação no Amazon SNS

Se você receber uma mensagem de erro informando que não tem autorização para executar uma ação, é preciso atualizar suas políticas para permitir que você realize a ação.

O erro do exemplo a seguir ocorre quando o usuário `mateojackson` tenta usar o console para visualizar detalhes sobre um recurso do *my-example-widget* fictício, mas não tem as permissões fictícias do `sns:GetWidget`.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
sns:GetWidget on resource: my-example-widget
```

Nesse caso, a política de Mateo deve ser atualizada para permitir que ele tenha acesso ao recurso *my-example-widget* usando a ação `sns:GetWidget`.

Se você precisar de ajuda, entre em contato com seu administrador da AWS. Seu administrador é a pessoa que forneceu suas credenciais de login.

## Não estou autorizado a realizar iam: PassRole

Caso receba uma mensagem de erro informando que você não tem autorização para executar a ação `iam:PassRole`, as políticas deverão ser atualizadas para permitir a transmissão de um perfil ao Amazon SNS.

Alguns Serviços da AWS permitem que você passe um perfil existente para o serviço, em vez de criar um novo perfil de serviço ou perfil vinculado ao serviço. Para fazer isso, é preciso ter permissões para passar o perfil para o serviço.

O exemplo de erro a seguir ocorre quando uma usuária do IAM chamada `marymajor` tenta usar o console para executar uma ação no Amazon SNS. No entanto, a ação exige que o serviço tenha permissões concedidas por um perfil de serviço. Mary não tem permissões para passar a função para o serviço.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
iam:PassRole
```

Nesse caso, as políticas de Mary devem ser atualizadas para permitir que ela realize a ação `iam:PassRole`.

Se você precisar de ajuda, entre em contato com seu administrador da AWS. Seu administrador é a pessoa que forneceu suas credenciais de login.

Quero permitir que as pessoas fora da minha Conta da AWS acessem meus recursos do Amazon SNS

É possível criar um perfil que os usuários de outras contas ou pessoas fora da sua organização possam utilizar para acessar seus recursos. Você pode especificar quem é confiável para assumir a função. Para serviços que oferecem suporte a políticas baseadas em recursos ou listas de controle de acesso (ACLs), você pode usar essas políticas para conceder às pessoas acesso aos seus recursos.

Para saber mais, consulte:

- Para saber se o Amazon SNS é compatível com esses recursos, consulte [Como o Amazon Simple Notification Service funciona com o IAM](#).
- Para saber como conceder acesso a seus atributos em todas as Contas da AWS pertencentes a você, consulte [Fornecimento de acesso a um usuário do IAM em outra Conta da AWS pertencente a você](#) no Guia de usuário do IAM.
- Para saber como conceder acesso a seus recursos para terceiros Contas da AWS, consulte [Fornecimento de acesso a Contas da AWS pertencentes a terceiros](#) no Guia do usuário do IAM.
- Para saber como conceder acesso por meio da federação de identidades, consulte [Conceder acesso a usuários autenticados externamente \(federação de identidades\)](#) no Guia do usuário do IAM.
- Para saber a diferença entre usar perfis e políticas baseadas em recursos para acesso entre contas, consulte [Como os perfis do IAM diferem de políticas baseadas em recursos](#) no Guia do usuário do IAM.

## Registrar em log e monitorar no Amazon SNS

Esta seção fornece informações sobre o monitoramento e o registro em log de tópicos do Amazon SNS.

Tópicos

- [Registrar em log chamadas de API do Amazon SNS usando o CloudTrail](#)
- [Monitorar tópicos do Amazon SNS usando o CloudWatch](#)

## Registrar em log chamadas de API do Amazon SNS usando o CloudTrail

O Amazon SNS é integrado ao AWS CloudTrail, um serviço que fornece um registro das ações tomadas por um usuário, uma função ou um serviço da AWS no Amazon SNS. O CloudTrail captura as chamadas de API para o Amazon SNS como eventos. As chamadas capturadas incluem as chamadas do console do Amazon SNS e as chamadas de código para as operações da API do Amazon SNS. Se você criar uma trilha, poderá habilitar a entrega contínua de eventos do CloudTrail para um bucket do Amazon S3, incluindo eventos para o Amazon SNS. Se você não configurar uma trilha, ainda poderá visualizar os eventos mais recentes no console do CloudTrail em Event history (Histórico de eventos). Usando as informações coletadas pelo CloudTrail, é possível determinar a solicitação feita para o Amazon SNS, o endereço IP do qual a solicitação foi feita, quem fez a solicitação, quando ela foi feita e detalhes adicionais.

Para saber mais sobre o CloudTrail, incluindo como configurá-lo e ativá-lo, consulte o [Guia do usuário do AWS CloudTrail](#).

### Informações sobre o Amazon SNS no CloudTrail

O CloudTrail é habilitado em sua Conta da AWS quando ela é criada. Quando a atividade do evento compatível ocorrer no Amazon SNS, ela será registrada em um evento do CloudTrail juntamente com outros eventos de serviços da AWS no Event history (Histórico de eventos). Você pode visualizar, pesquisar e baixar eventos recentes em sua Conta da AWS. Para obter mais informações, consulte [Visualizar eventos com o histórico de eventos do CloudTrail](#).

Para obter um registro de eventos em andamento na sua Conta da AWS, incluindo eventos do Amazon SNS, crie uma trilha. Uma trilha permite que o CloudTrail entregue arquivos de log a um bucket do Amazon S3. Por padrão, quando você cria uma trilha no console, ela é aplicada a todas as regiões da AWS. A trilha registra em log eventos de todas as regiões na partição da AWS e entrega os arquivos de log para o bucket do Amazon S3 especificado por você. Além disso, é possível configurar outros serviços da AWS para analisar mais ainda mais e agir com base nos dados de eventos coletados nos logs do CloudTrail. Para obter mais informações, consulte:

- [Visão geral da criação de uma trilha](#)
- [Serviços e integrações compatíveis com o CloudTrail](#)
- [Configurar notificações do Amazon SNS para o CloudTrail](#)
- [Receber arquivos de log do CloudTrail de várias regiões](#) e [receber arquivos de log do CloudTrail de várias contas](#)

## Eventos do ambiente de gerenciamento no CloudTrail

O Amazon SNS é compatível com as seguintes ações como eventos nos arquivos de log do CloudTrail:

- [AddPermission](#)
- [CheckIfPhoneNumberIsOptedOut](#)
- [ConfirmSubscription](#)
- [CreatePlatformApplication](#)
- [CreatePlatformEndpoint](#)
- [CreateSMSSandboxPhoneNumber](#)
- [CreateTopic](#)
- [DeleteEndpoint](#)
- [DeletePlatformApplication](#)
- [DeleteSMSSandboxPhoneNumber](#)
- [DeleteTopic](#)
- [GetDataProtectionPolicy](#)
- [GetEndpointAttributes](#)
- [GetPlatformApplicationAttributes](#)
- [GetSMSAttributes](#)
- [GetSMSSandboxAccountStatus](#)
- [GetSubscriptionAttributes](#)
- [GetTopicAttributes](#)
- [ListEndpointsByPlatformApplication](#)
- [ListOriginationNumbers](#)
- [ListPhoneNumbersOptedOut](#)
- [ListPlatformApplications](#)
- [ListSMSSandboxPhoneNumbers](#)
- [ListSubscriptions](#)
- [ListSubscriptionsByTopic](#)

- [ListTagsForResource](#)
- [ListTopics](#)
- [OptInPhoneNumber](#)
- [PutDataProtectionPolicy](#)
- [RemovePermission](#)
- [SetEndpointAttributes](#)
- [SetPlatformApplicationAttributes](#)
- [SetSMSAttributes](#)
- [SetSubscriptionAttributes](#)
- [SetTopicAttributes](#)
- [Subscribe](#)
- [TagResource](#)
- [Unsubscribe](#)
- [UntagResource](#)
- [VerifySMSSandboxPhoneNumber](#)

 Note

Se você não estiver conectado com uma conta da Amazon Web Services (modo não autenticado) e as ações [ConfirmSubscription](#) ou [Unsubscribe](#) forem chamadas, elas não serão registradas no CloudTrail. Por exemplo, quando você escolhe o link fornecido em uma notificação por e-mail para confirmar uma assinatura pendente em um tópico, a ação `ConfirmSubscription` é chamada no modo não autenticado. Neste exemplo, a ação `ConfirmSubscription` não seria registrada em log no CloudTrail.

Cada entrada de log ou evento contém informações sobre quem gerou a solicitação. As informações de identidade ajudam a determinar:

- Se a solicitação foi feita com credenciais de usuário raiz ou do AWS Identity and Access Management (IAM).
- Se a solicitação foi feita com credenciais de segurança temporárias de uma função ou de um usuário federado.

- Se a solicitação foi feita por outro serviço da AWS.

Para obter mais informações, consulte o [Elemento userIdentity do CloudTrail](#).

## Eventos do plano de dados no CloudTrail

Para habilitar o registro das ações de API a seguir nos arquivos do CloudTrail, você precisará ativar o registro de atividades da API do plano de dados no CloudTrail. Para obter mais informações, consulte [Registrar eventos de dados](#), no Guia do usuário do AWS CloudTrail.

Os eventos do plano de dados também podem ser filtrados por tipo de recurso para permitir um controle detalhado sobre quais chamadas de API do Amazon SNS você deseja registrar seletivamente em log e pagar no CloudTrail. Por exemplo, ao especificar `AWS::SNS::Topic` como um tipo de recurso, você pode registrar chamadas para ações de API `Publish` e `PublishBatch` referentes a tópicos. Da mesma forma, ao especificar `AWS::SNS::PlatformEndpoint` como um tipo de recurso, você pode registrar chamadas para a ação da API `Publish` referente a endpoints da plataforma. Para obter mais informações, consulte [AdvancedEventSelector](#) na Referência da API do AWS CloudTrail.

### Note

O tipo de recurso `AWS::SNS::PhoneNumber` do Amazon SNS não é registrado pelo CloudTrail.

## APIs de plano de dados do Amazon SNS

- [Publish](#)
- [PublishBatch](#)

## Exemplo: entradas de arquivo de log do Amazon SNS

Uma trilha é uma configuração que permite a entrega de eventos como arquivos de log a um bucket do Amazon S3 especificado. Os arquivos de log do CloudTrail contêm uma ou mais entradas de log. Um evento representa uma única solicitação de qualquer fonte e inclui informações sobre a ação solicitada, a data e a hora da ação, os parâmetros de solicitação e assim por diante. Os arquivos de log do CloudTrail não são um rastreamento de pilha ordenada de chamadas de API pública. Dessa forma, eles não são exibidos em uma ordem específica.

O exemplo a seguir mostra uma entrada de log do CloudTrail que demonstra as ações `ListTopics`, `CreateTopic` e `DeleteTopic`.

```
{
  "Records": [
    {
      "eventVersion": "1.02",
      "userIdentity": {
        "type": "IAMUser",
        "userName": "Bob",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::123456789012:user/Bob",
        "accountId": "123456789012",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE"
      },
      "eventTime": "2014-09-30T00:00:00Z",
      "eventSource": "sns.amazonaws.com",
      "eventName": "ListTopics",
      "awsRegion": "us-west-2",
      "sourceIPAddress": "127.0.0.1",
      "userAgent": "aws-sdk-java/unknown-version",
      "requestParameters": {
        "nextToken": "ABCDEF1234567890EXAMPLE=="
      },
      "responseElements": null,
      "requestID": "example1-b9bb-50fa-abdb-80f274981d60",
      "eventID": "example0-09a3-47d6-a810-c5f9fd2534fe",
      "eventType": "AwsApiCall",
      "recipientAccountId": "123456789012"
    },
    {
      "eventVersion": "1.02",
      "userIdentity": {
        "type": "IAMUser",
        "userName": "Bob",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::123456789012:user/Bob",
        "accountId": "123456789012",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE"
      },
      "eventTime": "2014-09-30T00:00:00Z",
      "eventSource": "sns.amazonaws.com",
      "eventName": "CreateTopic",
```

```
"awsRegion": "us-west-2",
"sourceIPAddress": "127.0.0.1",
"userAgent": "aws-sdk-java/unknown-version",
"requestParameters": {
  "name": "hello"
},
"responseElements": {
  "topicArn": "arn:aws:sns:us-west-2:123456789012:hello-topic"
},
"requestID": "example7-5cd3-5323-8a00-f1889011fee9",
"eventID": "examplec-4f2f-4625-8378-130ac89660b1",
"eventType": "AwsApiCall",
"recipientAccountId": "123456789012"
},
{
  "eventVersion": "1.02",
  "userIdentity": {
    "type": "IAMUser",
    "userName": "Bob",
    "principalId": "EX_PRINCIPAL_ID",
    "arn": "arn:aws:iam::123456789012:user/Bob",
    "accountId": "123456789012",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE"
  },
  "eventTime": "2014-09-30T00:00:00Z",
  "eventSource": "sns.amazonaws.com",
  "eventName": "DeleteTopic",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "127.0.0.1",
  "userAgent": "aws-sdk-java/unknown-version",
  "requestParameters": {
    "topicArn": "arn:aws:sns:us-west-2:123456789012:hello-topic"
  },
  "responseElements": null,
  "requestID": "example5-4faa-51d5-aab2-803a8294388d",
  "eventID": "example8-6443-4b4d-abfd-1b867280d964",
  "eventType": "AwsApiCall",
  "recipientAccountId": "123456789012"
},
]
}
```

Os exemplos a seguir mostram entradas de log do CloudTrail que demonstram as ações Publish e PublishBatch.

## Publicar

```
{
  "eventVersion": "1.09",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "EX_PRINCIPAL_ID",
    "arn": "arn:aws:iam::123456789012:user/Bob",
    "accountId": "123456789012",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "sessionContext": {
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AKIAIOSFODNN7EXAMPLE",
        "arn": "arn:aws:iam::123456789012:role/Admin",
        "accountId": "123456789012",
        "userName": "ExampleUser"
      },
      "attributes": {
        "creationDate": "2023-08-21T16:44:05Z",
        "mfaAuthenticated": "false"
      }
    }
  },
  "eventTime": "2023-08-21T16:48:37Z",
  "eventSource": "sns.amazonaws.com",
  "eventName": "Publish",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "192.0.2.0",
  "userAgent": "aws-cli/1.29.16 md/Botocore#1.31.16 ua/2.0 os/
linux#5.4.250-173.369.amzn2int.x86_64 md/arch#x86_64 lang/python#3.8.17 md/
pyimpl#CPython cfg/retry-mode#legacy botocore/1.31.16",
  "requestParameters": {
    "topicArn": "arn:aws:sns:us-east-1:123456789012:ExampleSNSTopic",
    "message": "HIDDEN_DUE_TO_SECURITY_REASONS",
    "subject": "HIDDEN_DUE_TO_SECURITY_REASONS",
    "messageStructure": "json",
    "messageAttributes": "HIDDEN_DUE_TO_SECURITY_REASONS"
  },
  "responseElements": {
    "messageId": "0787cd1e-d92b-521c-a8b4-90434e8ef840"
  }
}
```

```
},
"requestID": "0a8ab208-11bf-5e01-bd2d-ef55861b545d",
"eventID": "bb3496d4-5252-4660-9c28-3c6aebdb21c0",
"readOnly": false,
"resources": [{
  "accountId": "123456789012",
  "type": "AWS::SNS::Topic",
  "ARN": "arn:aws:sns:us-east-1:123456789012:ExampleSNSTopic"
}],
"eventType": "AwsApiCall",
"managementEvent": false,
"recipientAccountId": "123456789012",
"eventCategory": "Data",
"tlsDetails": {
  "tlsVersion": "TLSv1.2",
  "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
  "clientProvidedHostHeader": "sns.us-east-1.amazonaws.com"
}
}
```

## PublishBatch

```
{
  "eventVersion": "1.09",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "EX_PRINCIPAL_ID",
    "arn": "arn:aws:iam::123456789012:user/Bob",
    "accountId": "123456789012",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "sessionContext": {
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AKIAIOSFODNN7EXAMPLE",
        "arn": "arn:aws:iam::123456789012:role/Admin",
        "accountId": "123456789012",
        "userName": "ExampleUser"
      },
    },
    "attributes": {
      "creationDate": "2023-08-21T19:20:49Z",
      "mfaAuthenticated": "false"
    }
  }
}
```

```
},
"eventTime": "2023-08-21T19:22:01Z",
"eventSource": "sns.amazonaws.com",
"eventName": "PublishBatch",
"awsRegion": "us-east-1",
"sourceIPAddress": "192.0.2.0",
"userAgent": "aws-cli/1.29.16 md/Botocore#1.31.16 ua/2.0 os/
linux#5.4.250-173.369.amzn2int.x86_64 md/arch#x86_64 lang/python#3.8.17 md/
pyimpl#CPython cfg/retry-mode#legacy botocore/1.31.16",
"requestParameters": {
  "topicArn": "arn:aws:sns:us-east-1:123456789012:ExampleSNSTopic",
  "publishBatchRequestEntries": [{
    "id": "1",
    "message": "HIDDEN_DUE_TO_SECURITY_REASONS"
  },
  {
    "id": "2",
    "message": "HIDDEN_DUE_TO_SECURITY_REASONS"
  }
]
},
"responseElements": {
  "successful": [{
    "id": "1",
    "messageId": "30d68101-a64a-5573-9e10-dc5c1dd3af2f"
  },
  {
    "id": "2",
    "messageId": "c0aa0c5c-561d-5455-b6c4-5101ed84de09"
  }
],
  "failed": []
},
"requestID": "e2cdf7f3-1b35-58ad-ac9e-aaaaea0ace2f1",
"eventID": "10da9a14-0154-4ab6-b3a5-1825b229a7ed",
"readOnly": false,
"resources": [{
  "accountId": "123456789012",
  "type": "AWS::SNS::Topic",
  "ARN": "arn:aws:sns:us-east-1:123456789012:ExampleSNSTopic"
}],
"eventType": "AwsApiCall",
"managementEvent": false,
"recipientAccountId": "123456789012",
```

```
"eventCategory": "Data",
"tlsDetails": {
  "tlsVersion": "TLSv1.2",
  "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
  "clientProvidedHostHeader": "sns.us-east-1.amazonaws.com"
}
}
```

## Monitorar tópicos do Amazon SNS usando o CloudWatch

O Amazon SNS e o Amazon CloudWatch são integrados para que você possa coletar, visualizar e analisar métricas para cada notificação ativa do Amazon SNS. Após configurar o CloudWatch para o Amazon SNS, você pode obter mais informações sobre a performance dos tópicos, notificações por push e entregas de SMS do Amazon SNS. Por exemplo, você pode definir um alarme para enviar uma notificação por e-mail se um limite especificado para uma métrica do Amazon SNS for atingido, como `NumberOfNotificationsFailed`. Para obter uma lista de todas as métricas enviadas pelo Amazon SNS para o CloudWatch, consulte [Métricas do Amazon SNS](#). Para obter mais informações sobre notificações por push do Amazon SNS, consulte [Notificações por push para dispositivos móveis](#).

### Note

As métricas configuradas com o CloudWatch para os tópicos do Amazon SNS são coletadas e enviadas automaticamente ao CloudWatch em intervalos de 1 minuto. Essas métricas são coletadas em todos os tópicos que estejam de acordo com as diretrizes do CloudWatch por estarem ativas. Um tópico é considerado ativo pelo CloudWatch por até seis horas desde a última atividade (ou seja, qualquer chamada de API) no tópico.

Não há cobrança para as métricas do Amazon SNS relatadas no CloudWatch. Elas são fornecidas como parte do serviço Amazon SNS.

## Visualizar métricas do CloudWatch relacionadas ao Amazon SNS

Você pode monitorar as métricas do Amazon SNS usando o console do CloudWatch, a própria interface de linha de comando (CLI) do CloudWatch ou de forma programática usando a API do CloudWatch. Os procedimentos a seguir mostram como acessar as métricas com o AWS Management Console.

## Como visualizar métricas usando o console do CloudWatch

1. Faça login no [console do CloudWatch](#).
2. No painel de navegação, selecione Métricas.
3. Na guia All metrics (Todas as métricas), escolha SNS e uma das seguintes dimensões:
  - Country, SMS Type (País, Tipo de SMS)
  - PhoneNumber
  - Topic Metrics (Métricas de tópico)
  - Metrics with no dimensions (Métricas sem dimensões)
4. Para exibir mais detalhes, escolha um item específico. Por exemplo, se você escolher Topic Metrics (Métricas de tópico) e NumberOfMessagesPublished, o número médio de mensagens do Amazon SNS publicadas por um período de um minuto ao longo do intervalo de seis horas será exibido.
5. Para visualizar as métricas de uso do Amazon SNS, na guia All metrics (Todas as métricas), escolha Usage (Uso) e selecione a target Amazon SNS usage metric (métrica-alvo de uso do Amazon SNS) (por exemplo, NumberOfMessagesPublishedPerAccount).

## Definir alarmes do CloudWatch para métricas do Amazon SNS

O CloudWatch também permite que você defina alarmes que serão acionados quando o limite de uma métrica for atingido. Por exemplo, você pode definir um alarme para a métrica, NumberOfNotificationsFailed, de modo que, quando o número limite especificado for atendido dentro do período de amostragem, uma notificação será enviada por e-mail para informá-lo sobre o evento.

Para definir alertas usando o console do CloudWatch

1. Faça login no AWS Management Console e abra o console do CloudWatch em <https://console.aws.amazon.com/cloudwatch/>.
2. Selecione Alarmes e clique no botão Criar alarme. Isso inicia o assistente Criar alarme.
3. Role pelas métricas do Amazon SNS para localizar aquela em que você deseja colocar um alarme. Selecione a métrica para criar um alarme e escolha Continue (Continuar).
4. Preencha os valores de Name (Nomes), Description (Descrição), Threshold (Limite) e Time (Tempo) para a métrica e escolha Continue (Continuar).
5. Escolha Alarme como o estado do alarme. Para que o CloudWatch envie um e-mail quando o estado de alarme for atingido, selecione um tópico existente do Amazon SNS ou escolha Create

New Email Topic (Criar novo tópico de e-mail). Se você escolher Create New Email Topic (Criar novo tópico de e-mail), poderá definir o nome e os endereços de e-mail para um novo tópico. Esta lista será salva e aparecerá na caixa suspensa para alertas futuros. Escolha Continue (Continuar).

#### Note

Se você escolher Create New Email Topic (Criar novo tópico de e-mail) para criar um novo tópico do Amazon SNS, os endereços de e-mail deverão ser verificados para que possam receber notificações. Os e-mails são enviados somente quando o alerta entra em um estado de alerta. Se essa alteração para estado de alerta ocorrer antes que os endereços de e-mail sejam verificados, a notificação não será recebida.

- Nesse momento, o assistente Criar alarme lhe oferece uma oportunidade para revisar o alarme que você está prestes a criar. Se você precisar fazer alterações, use os links Editar à direita. Quando estiver satisfeito, escolha Create Alarm (Criar alarme).

Para obter mais informações sobre o uso do CloudWatch e dos alarmes, consulte a [Documentação do CloudWatch](#).

## Métricas do Amazon SNS

O Amazon SNS envia as métricas a seguir ao CloudWatch.

Namespace	Métrica	Descrição
AWS/SNS	NumberOfMessagesPublished	<p>O número de mensagens publicadas nos tópicos do Amazon SNS.</p> <p>Unidades: Contagem</p> <p>Dimensões válidas: Application, PhoneNumber, Platform e TopicName</p> <p>Estatísticas válidas: soma</p>

Namespace	Métrica	Descrição
AWS/SNS	NumberOfNotificationsDelivered	<p>O número de mensagens entregues com êxito dos tópicos do Amazon SNS para endpoints de assinatura.</p> <p>Para obter uma tentativa de entrega bem-sucedida, o endpoint deve aceitar a assinatura da mensagem. Uma assinatura aceitará uma mensagem se a.) faltar uma política de filtro ou b.) a política de filtro incluir atributos que combinem com os atribuídos à mensagem. Se a assinatura rejeita a mensagem, a tentativa de entrega não é contada para esta métrica.</p> <p>Unidades: Contagem</p> <p>Dimensões válidas: Application, PhoneNumber, Platform e TopicName</p> <p>Estatísticas válidas: soma</p>

Namespace	Métrica	Descrição
AWS/SNS	NumberOfNotificationsFailed	<p>O número de mensagens do Amazon SNS com falha na entrega.</p> <p>Para o Amazon SQS, e-mail, SMS ou endpoints de push para dispositivos móveis, a métrica é incrementada em 1 quando o Amazon SNS tenta entregar as mensagens. Para endpoints HTTP ou HTTPS, a métrica inclui cada tentativa de entrega com falha, incluindo novas tentativas que seguem a tentativa inicial. Para todos os outros endpoints, a contagem aumenta em 1 quando a mensagem não é entregue (independentemente do número de tentativas).</p> <p>Essa métrica não inclui mensagens que foram rejeitadas pelo filtro de assinatura políticas.</p> <p>Você pode controlar o número de novas tentativas para endpoints HTTP. Para obter mais informações, consulte <a href="#">Novas tentativas de entrega de mensagens do Amazon SNS</a>.</p> <p>Unidades: Contagem</p> <p>Dimensões válidas: Application, PhoneNumber, Platform e TopicName</p>

Namespace	Métrica	Descrição
		Estatísticas válidas: soma, média
AWS/SNS	NumberOfNotificationsFilteredOut	<p>O número de mensagens que foram rejeitadas pelo filtro de assinatura políticas. Uma política de filtro rejeita uma mensagem quando os atributos de mensagem não correspondem aos atributos de política.</p> <p>Unidades: Contagem</p> <p>Dimensões válidas: Application, PhoneNumber, Platform e TopicName</p> <p>Estatísticas válidas: soma, média</p>
AWS/SNS	NumberOfNotificationsFilteredOut-MessageAttributes	<p>O número de mensagens que foram rejeitadas por políticas de filtro de assinatura para filtragem baseada em atributos.</p> <p>Unidades: CountValid</p> <p>Dimensões: Application, PhoneNumber, Platform e TopicName</p> <p>Estatísticas válidas: soma, média</p>

Namespace	Métrica	Descrição
AWS/SNS	NumberOfNotificationsFilteredOut-MessageBody	<p>O número de mensagens que foram rejeitadas por políticas de filtro de assinatura para filtragem baseada em carga útil.</p> <p>Unidade: contagem</p> <p>Dimensões válidas: Application, PhoneNumber, Platform e TopicName</p> <p>Estatísticas válidas: soma, média</p>
AWS/SNS	NumberOfNotificationsFilteredOut-InvalidAttributes	<p>O número de mensagens que foram rejeitadas por políticas de filtro de assinatura por conta de atributos de mensagens são inválidos. Por exemplo, devido à formatação incorreta do atributo JSON.</p> <p>Unidades: Contagem</p> <p>Dimensões válidas: Application, PhoneNumber, Platform e TopicName</p> <p>Estatísticas válidas: soma, média</p>

Namespace	Métrica	Descrição
AWS/SNS	NumberOfNotificationsFilteredOut-NoMessageAttributes	<p>O número de mensagens que foram rejeitadas pelas políticas de filtro de assinatura por conta de as mensagens não terem atributos.</p> <p>Unidades: Contagem</p> <p>Dimensões válidas: Application, PhoneNumber, Platform e TopicName</p> <p>Estatísticas válidas: soma, média</p>
AWS/SNS	NumberOfNotificationsFilteredOut-InvalidMessageBody	<p>O número de mensagens que foram rejeitadas por políticas de filtro de assinatura porque o corpo da mensagem era inválido para filtragem; por exemplo, corpo da mensagem JSON inválido.</p> <p>Unidade: contagem</p> <p>Dimensões válidas: Application, PhoneNumber, Platform e TopicName</p> <p>Estatísticas válidas: soma, média</p>

Namespace	Métrica	Descrição
AWS/SNS	NumberOfNotificationsRedrivenToDlq	<p>O número de mensagens que foram movidas para uma fila de mensagens mortas.</p> <p>Unidades: Contagem</p> <p>Dimensões válidas: Application, PhoneNumber, Platform e TopicName</p> <p>Estatísticas válidas: soma, média</p>
AWS/SNS	NumberOfNotificationsFailedToRedriveToDlq	<p>O número de mensagens que não puderam ser movidas para uma fila de mensagens mortas.</p> <p>Unidades: Contagem</p> <p>Dimensões válidas: Application, PhoneNumber, Platform e TopicName</p> <p>Estatísticas válidas: soma, média</p>
AWS/SNS	PublishSize	<p>O tamanho das mensagens publicadas.</p> <p>Unidade: bytes</p> <p>Dimensões válidas: Application, PhoneNumber, Platform e TopicName</p> <p>Estatísticas válidas: mínimo, máximo, média e contagem</p>

Namespace	Métrica	Descrição
AWS/SNS	SMSMonthToDateSpentUSD	<p>As cobranças que você acumulou desde o início do mês atual com o envio de mensagens SMS.</p> <p>Você pode definir um alarme para esta métrica para saber quando seus encargos acumulados do mês atual estão próximos da cota de gasto mensal de SMS da sua conta. Quando o Amazon SNS determina que o envio de uma mensagem SMS pode gerar um custo que excede essa cota, ele interrompe a publicação de mensagens SMS em poucos minutos.</p> <p>Para obter informações sobre como configurar sua cota de gasto mensal de SMS ou para obter informações sobre como solicitar um aumento de cota de gasto com a AWS, consulte <a href="#">Definir preferências de mensagens SMS</a>.</p> <p>Unidade: USD</p> <p>Dimensões válidas: PhoneNumber</p> <p>Estatísticas válidas: máximo</p>

Namespace	Métrica	Descrição
AWS/SNS	SMSSuccessRate	<p>A taxa de entregas bem-sucedidas de mensagem SMS.</p> <p>Unidades: Contagem</p> <p>Dimensões válidas: PhoneNumber</p> <p>Estatísticas válidas: soma, média, amostras de dados</p>

## Dimensões para métricas do Amazon SNS

O Amazon Simple Notification Service envia as seguintes dimensões ao CloudWatch.

Dimensão	Descrição
Application	Os filtros em objetos da aplicação, que representam uma aplicação e o dispositivo registrado com um dos serviços de notificação por push compatíveis, como as APNs e o FCM.
Application, Platform	Os filtros em objetos da aplicação e da plataforma, em que os objetos da plataforma são destinados aos serviços de notificação por push compatíveis, como as APNs e o FCM.
Country	Os filtros no país ou região de destino de uma mensagem SMS. O país ou região é representado por seu código alfa-2 ISO 3166-1.
PhoneNumber	Filtra pelo número de telefone quando o SMS é publicado diretamente em um número de telefone (sem um tópico).
Platform	Filtra pelos objetos da plataforma destinados aos serviços de notificação por push, como as APNs e o FCM.
TopicName	Filtra pelos nomes dos tópicos do Amazon SNS.

Dimensão	Descrição
SMSType	Os filtros no tipo da mensagem SMS. Podem ser promocionais ou transacionais.

## Métricas de uso do Amazon SNS

O Amazon Simple Notification Service envia as seguintes métricas de uso ao CloudWatch.

Namespace	Serviço	Métrica	Recurso	Tipo	Descrição
AWS/Usage	SNS	ResourceCount	NumberOfMessagesPublishedPerAccount	Recurso	<ul style="list-style-type: none"> <li>O número de mensagens publicadas nos tópicos do Amazon SNS em sua conta da AWS.</li> <li>Unidades: nenhuma</li> <li>Estatísticas válidas: soma</li> </ul>
AWS/Usage	SNS	ResourceCount	ApproximateNumberOfTopics	Recurso	<ul style="list-style-type: none"> <li>O número aproximado de tópicos em sua conta da AWS.</li> <li>Unidades: nenhuma</li> <li>Estatísticas válidas:</li> </ul>

Namespace	Serviço	Métrica	Recurso	Tipo	Descrição
					média, mínimo, máximo, soma
AWS/Usage	SNS	ResourceCount	ApproximateNumberOfFilterPolicies	Recurso	<ul style="list-style-type: none"><li>• O número aproximado de políticas de filtro em sua conta da AWS.</li><li>• Unidades: nenhuma</li><li>• Estatísticas válidas: média, mínimo, máximo, soma</li></ul>

Namespace	Serviço	Métrica	Recurso	Tipo	Descrição
AWS/Usage	SNS	ResourceCount	ApproximateNumberOfPendingSubscriptions	Recurso	<ul style="list-style-type: none"><li>• O número aproximado de assinaturas pendentes em sua conta da AWS.</li><li>• Unidades: nenhuma</li><li>• Estatísticas válidas: média, mínimo, máximo, soma</li></ul>

Namespace	Serviço	Métrica	Recurso	Tipo	Descrição
AWS/Usage	SNS	CallCount	<ul style="list-style-type: none"> <li>AddPermission</li> <li>CheckIfPhoneNumberIsOptedOut</li> <li>CreatePlatformApplication</li> <li>CreatePlatformEndpoint</li> <li>ConfirmSubscription</li> <li>CreateSMSSandboxPhoneNumber</li> <li>CreateTopic</li> <li>DeleteEndpoint</li> <li>DeletePlatformApplication</li> <li>DeleteSMSSandboxPhoneNumber</li> <li>DeleteTopic</li> </ul>	API	<ul style="list-style-type: none"> <li>O número de chamadas de API para a API do Amazon SNS selecionada em sua conta da AWS.</li> <li>Unidades: nenhuma</li> <li>Estatísticas válidas: soma</li> </ul>

Namespace	Serviço	Métrica	Recurso	Tipo	Descrição
			<ul style="list-style-type: none"><li>• <code>GetEndpointAttributes</code></li><li>• <code>GetPlatformApplicationAttributes</code></li><li>• <code>GetSMSAttributes</code></li><li>• <code>GetSMSSandboxAccountStatus</code></li><li>• <code>GetSubscriptionAttributes</code></li><li>• <code>GetTopicAttributes</code></li> <li>• <code>ListEndpointsByPlatformApplication</code></li><li>• <code>ListOriginNumbers</code></li><li>• <code>ListPhoneNumbersOptedOut</code></li><li>• <code>ListPlatformApplications</code></li></ul>		

Namespace	Serviço	Métrica	Recurso	Tipo	Descrição
			<ul style="list-style-type: none"><li>ListSMSSandboxPhoneNumbers</li><li>ListSubscriptions</li><li>ListSubscriptionsByTopic</li><li>ListTagsForResource</li><li>ListTopics</li><li>OptInPhoneNumber</li><li>RemovePermission</li><li>SetEndpointAttributes</li><li>SetPlatformApplicationAttributes</li><li>SetSMSAttributes</li><li>SetSubscriptionAttributes</li><li>SetTopicAttributes</li></ul>		

Namespace	Serviço	Métrica	Recurso	Tipo	Descrição
			<ul style="list-style-type: none"> <li>Subscribe</li> <li>Unsubscribe</li> <li>UntagResource</li> <li>VerifySMSSandboxPhoneNumber</li> </ul>		

## Validação de conformidade para o Amazon SNS

Audidores de terceiros avaliam a segurança e a conformidade do Amazon SNS como parte de vários programas de conformidade da AWS, incluindo a Lei de Portabilidade e Responsabilidade de Seguros de Saúde (HIPAA).

Para obter uma lista de serviços da AWS no escopo de programas de conformidade específicos, consulte [Serviços da AWS no escopo por programa de conformidade](#). Para obter informações gerais, consulte [Programas de conformidade da AWS](#).

É possível fazer download de relatórios de auditoria de terceiros usando o AWS Artifact. Para obter mais informações, consulte [Fazer download de relatórios no AWS Artifact](#).

Sua responsabilidade com relação à compatibilidade ao usar o Amazon SNS é determinada pelo sigilo dos seus dados, pelos objetivos de compatibilidade da sua empresa e pelas leis e regulamentos aplicáveis. A AWS fornece os seguintes recursos para ajudar na compatibilidade:

- [Guias de início rápido de segurança e compatibilidade](#): estes guias de implantação abordam as considerações de arquitetura e fornecem etapas para implantação de ambientes de linha de base focados em compatibilidade e segurança na AWS.
- [Whitepaper Architecting for HIPAA Security and Compliance](#): este whitepaper descreve como as empresas podem usar a AWS para criar aplicações em conformidade com a HIPAA.
- [Recursos de conformidade da AWS](#): essa coleção de manuais e guias pode ser aplicada a seu setor e local.

- [Avaliar recursos com regras](#) no AWS Config Developer Guide (Guia do desenvolvedor do CCI): o serviço AWS Config avalia como as configurações de recursos estão em conformidade com práticas internas, diretrizes do setor e regulamentos.
- [AWS Security Hub](#): esse serviço da AWS fornece uma visão abrangente do estado de sua segurança na AWS que ajuda você a conferir sua conformidade com padrões e práticas recomendadas de segurança do setor.

## Resiliência no Amazon SNS

A resiliência no Amazon SNS é garantida por meio da utilização AWS da infraestrutura global, que gira em torno Regiões da AWS de zonas de disponibilidade. Regiões da AWS oferecem zonas de disponibilidade fisicamente separadas e isoladas conectadas por redes de baixa latência, alto rendimento e altamente redundantes. Essa arquitetura permite um failover contínuo entre as zonas de disponibilidade sem interrupção, tornando os aplicativos e bancos de dados inerentemente mais tolerantes a falhas e escaláveis em comparação com as infraestruturas tradicionais de data center. Ao usar zonas de disponibilidade, os assinantes do Amazon SNS se beneficiam de maior disponibilidade e confiabilidade, garantindo a entrega de mensagens apesar de possíveis interrupções. Para obter mais informações sobre zonas de disponibilidade Regiões da AWS e zonas de disponibilidade, consulte [Infraestrutura AWS global](#).

Além disso, as assinaturas de tópicos do Amazon SNS podem ser configuradas com novas tentativas de entrega e filas de cartas mortas, permitindo o tratamento automático de falhas transitórias e garantindo que as mensagens cheguem de forma confiável aos destinos pretendidos.

O Amazon SNS também suporta filtragem de mensagens e atributos de mensagens, o que permite adaptar estratégias de resiliência a seus casos de uso específicos, aprimorando a robustez geral de seus aplicativos.

## Segurança da infraestrutura no Amazon SNS

Como um serviço gerenciado, o Amazon SNS é protegido pelos procedimentos AWS globais de segurança de rede encontrados na documentação de [melhores práticas de segurança, identidade e conformidade](#).

Use ações de AWS API para acessar o Amazon SNS pela rede. Os clientes devem oferecer suporte a Transport Layer Security (TLS) 1.2 ou posterior. Os clientes também devem ter suporte a pacotes

de criptografia com sigilo de encaminhamento perfeito (PFS) como Ephemeral Diffie-Hellman (DHE) ou Ephemeral Elliptic Curve Diffie-Hellman (ECDHE).

Você deve assinar as solicitações usando um ID de chave de acesso e uma chave de acesso secreta associados a uma entidade principal do IAM. Como alternativa, você pode usar o [AWS Security Token Service](#) (AWS STS) para gerar credenciais de segurança temporárias para assinar solicitações.

Você pode chamar essas ações de API de qualquer local da rede, mas o Amazon SNS oferece suporte a políticas de acesso com base em recursos, que podem incluir restrições com base no endereço IP de origem. Também é possível usar políticas do Amazon SNS para controlar o acesso de endpoints da Amazon VPC ou de VPCs específicas. Isso isola efetivamente o acesso à rede a um determinado tópico do Amazon SNS somente da VPC específica dentro da rede. Para ter mais informações, consulte [Restringir a publicação em um tópico do Amazon SNS somente de um endpoint da VPC específico](#).

## Práticas recomendadas de segurança para o Amazon SNS

A AWS fornece muitos recursos de segurança para o Amazon SNS. Analise esses recursos de segurança no contexto de sua própria política de segurança.

### Note

As orientações desses recursos de segurança se aplicam a casos de uso e implementações comuns. Recomendamos que você examine as melhores práticas no contexto do seu caso de uso, arquitetura e modelo de ameaças específicos.

## Melhores práticas preventivas

Veja a seguir as práticas recomendadas de segurança preventiva para o Amazon SNS.

### Tópicos

- [Verifique se os tópicos não estão acessíveis ao público geral](#)
- [Implemente o privilégio de acesso mínimo](#)
- [Usar funções do IAM para aplicações e produtos da AWS que exigem acesso ao Amazon SNS](#)
- [Implemente a criptografia no lado do servidor](#)

- [Aplicar a criptografia de dados em trânsito](#)
- [Considerar o uso de endpoints da VPC para acessar o Amazon SNS](#)
- [Certificar-se de que as assinaturas não estejam configuradas para entregar a endpoints http brutos](#)

## Verifique se os tópicos não estão acessíveis ao público geral

A menos que seja nitidamente necessário que qualquer pessoa na Internet possa ler ou escrever no tópico do Amazon SNS, você deve garantir que o tópico não seja acessível ao público (acessível por todo mundo ou por qualquer usuário autenticado da AWS).

- Evite criar políticas com o `Principal` definido como `"`.
- Evite usar um curinga (\*). Em vez disso, nomeie um usuário ou usuários específicos.

## Implemente o privilégio de acesso mínimo

Quando concede permissões, você decide quem as recebe, para quais tópicos as permissões se aplicam e as ações específicas de API que você deseja permitir para esses tópicos. A implementação do princípio de privilégio mínimo é importante para reduzir os riscos de segurança. Também ajuda a reduzir o efeito negativo de erros ou intenção maliciosa.

Siga o aviso de segurança padrão de concessão de privilégios mínimos. Ou seja, conceda apenas as permissões necessárias para executar uma tarefa específica. Você pode implementar o privilégio mínimo usando uma combinação de políticas de segurança pertencentes ao acesso do usuário.

O Amazon SNS usa o modelo publicador-assinante, exigindo três tipos de acesso à conta de usuário:

- Administradores: acesso para criar, modificar e excluir tópicos. Os administradores também controlam as políticas de tópicos.
- Editores: acesso ao envio de mensagens para tópicos.
- Assinantes: acesso à assinatura de tópicos.

Para obter mais informações, consulte as seções a seguir:

- [Gerenciamento de identidade e acesso no Amazon SNS](#)
- [Permissões da API do Amazon SNS: referência de ações e recursos](#)

## Usar funções do IAM para aplicações e produtos da AWS que exigem acesso ao Amazon SNS

Para aplicações ou serviços da AWS, como o Amazon EC2, para obter acesso aos tópicos do Amazon SNS, é preciso usar credenciais válidas da AWS nas solicitações de API da AWS. Como essas credenciais não são alteradas automaticamente, você não deve armazenar credenciais da AWS diretamente no aplicativo ou na instância do EC2.

Use uma função do IAM para gerenciar credenciais temporárias para aplicações ou serviços que precisam acessar o Amazon SNS. Quando você usa um perfil, não é necessário distribuir credenciais de longo prazo (como um nome de usuário, uma senha e chaves de acesso) para uma instância do EC2 ou um serviço da AWS como o AWS Lambda. Em vez disso, a função fornece permissões temporárias que os aplicativos podem usar ao fazer chamadas para outros recursos da AWS.

Para obter mais informações, consulte [IAM Roles](#) (Funções do IAM) e [Common Scenarios for Roles: Users, Applications, and Services](#) (Cenários comuns para funções: usuários, aplicações e produtos) no Guia do usuário do IAM.

### Implemente a criptografia no lado do servidor

Para atenuar problemas de vazamento de dados, utilize a criptografia em repouso para criptografar as mensagens usando uma chave armazenada em um local diferente do local de armazenamento das suas mensagens. A criptografia no lado do servidor (SSE) fornece criptografia dos dados em repouso. O Amazon SNS criptografa os dados no nível da mensagem ao armazená-los e descriptografa as mensagens para você, quando você as acessa. O SSE usa chaves gerenciadas pelo AWS Key Management Service. Quando você autentica sua solicitação e tem as permissões de acesso, não há diferença de acesso entre as filas criptografadas e não criptografadas.

Para obter mais informações, consulte [Criptografia inativa](#) e [Gerenciamento de chaves](#).

### Aplicar a criptografia de dados em trânsito

É possível, mas não recomendado, publicar mensagens que não estejam criptografadas durante o trânsito usando HTTP. No entanto, não é possível usar HTTP ao publicar em um tópico criptografado do SNS.

A AWS recomenda o uso de HTTPS em vez de HTTP. Ao usar HTTPS, as mensagens são criptografadas automaticamente durante o trânsito, mesmo que o tópico do SNS em si não esteja criptografado. Sem HTTPS, um invasor que venha pela rede pode bisbilhotar o tráfego de rede ou manipulá-lo usando um ataque, como man-in-the-middle.

Para impor somente conexões criptografadas por HTTPS, adicione a condição [aws:SecureTransport](#) à política do IAM anexada aos tópicos do SNS não criptografados. Isso força os editores de mensagens a usar HTTPS em vez de HTTP. Você pode usar o seguinte exemplo de política como um guia:

```
{
  "Id": "ExamplePolicy",
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowPublishThroughSSLOnly",
      "Action": "SNS:Publish",
      "Effect": "Deny",
      "Resource": [
        "arn:aws:sns:us-east-1:1234567890:test-topic"
      ],
      "Condition": {
        "Bool": {
          "aws:SecureTransport": "false"
        }
      },
      "Principal": "*"
    }
  ]
}
```

## Considerar o uso de endpoints da VPC para acessar o Amazon SNS

Se você tiver tópicos com os quais é preciso interagir, mas que não devem de forma alguma ficar expostos à Internet, use VPC endpoints para limitar o acesso ao tópico apenas aos hosts dentro de uma VPC específica. Você pode usar políticas de tópico para controlar o acesso a tópicos de endpoints da Amazon VPC ou a VPCs específicas.

Os endpoints da VPC do Amazon SNS fornecem duas maneiras de controlar o acesso às suas mensagens:

- É possível controlar as solicitações, os usuários ou os grupos permitidos por um VPC endpoint específico.
- Você pode controlar quais VPCs ou VPC endpoints terão acesso ao tópico usando uma política de tópico.

Para obter mais informações, consulte [Criar o endpoint](#) e [Criar uma política de endpoint da Amazon VPC para o Amazon SNS](#).

**Certificar-se de que as assinaturas não estejam configuradas para entregar a endpoints http brutos**

Evite configurar assinaturas para entregar a endpoints http brutos. Sempre tenha assinaturas entregando para um nome de domínio de endpoint. Por exemplo, uma assinatura configurada para entregar a um endpoint, `http://1.2.3.4/my-path`, deve ser alterada para `http://my.domain.name/my-path`.

# Solução de problemas com tópicos do Amazon SNS

Esta seção fornece informações sobre como solucionar problemas de tópicos do Amazon SNS.

## Solução de problemas com tópicos do Amazon SNS usando o AWS X-Ray

O AWS X-Ray coleta dados sobre solicitações que seu aplicativo atende e permite visualizar e filtrar dados para identificar possíveis problemas e oportunidades de otimização. Para qualquer solicitação rastreada para o aplicativo, é possível ver informações detalhadas sobre a solicitação, a resposta e as chamadas que o aplicativo faz para recursos, microsserviços, bancos de dados da AWS de downstream e APIs da web HTTP.

É possível usar o X-Ray com o Amazon SNS para rastrear e analisar as mensagens que passam pela aplicação. É possível usar o AWS Management Console para visualizar o mapa de conexões entre o Amazon SNS e outros serviços que a aplicação usa. Também é possível usar o console para visualizar métricas como a latência média e as taxas de falha. Para obter mais informações, consulte [Amazon SNS e AWS X-Ray](#) no Guia do desenvolvedor do AWS X-Ray.

## Rastreamento ativo no Amazon SNS

[Você pode usar AWS X-Ray para rastrear e analisar solicitações de usuários à medida que elas percorrem seus tópicos do Amazon SNS até suas assinaturas de endpoint do Amazon Data Firehose, Amazon AWS LambdaSQS e HTTP/S.](#) Como o X-Ray fornece uma end-to-end visão de uma solicitação completa, você pode ver o que está chamando seu tópico do Amazon SNS e o que está abaixo das assinaturas do seu tópico. Você pode analisar as latências em suas mensagens e em seus serviços de back-end (por exemplo, quanto tempo uma solicitação passa em um tópico e quanto tempo levou para entregar a mensagem a cada uma das assinaturas do tópico).

### Important

Os tópicos do Amazon SNS com várias assinaturas podem atingir um limite de tamanho e não ser totalmente rastreados. Para obter informações sobre os limites de tamanho do documento de rastreamento, consulte [Cotas de serviço do X-Ray](#) na Referência geral da AWS.

Se você chamar uma API do Amazon SNS de um serviço que já foi rastreado, o Amazon SNS enviará o rastreamento adiante, mesmo que o rastreamento do X-Ray não esteja habilitado na API.

O Amazon SNS permite o rastreamento do X-Ray para tópicos comuns e FIFO. Você pode ativar o X-Ray para um tópico do Amazon SNS usando o [console do Amazon SNS](#), a [API SetTopicAttributes do Amazon SNS](#), a [referência de CLI do Amazon Simple Notification Service](#) ou o [AWS CloudFormation](#).

Para saber mais sobre como usar o Amazon SNS com o X-Ray, consulte [Amazon SNS e AWS X-Ray](#) no Guia do desenvolvedor do AWS X-Ray.

## Tópicos

- [Permissões de rastreamento ativo](#)
- [Habilitar o rastreamento ativo em um tópico do Amazon SNS \(console\)](#)
- [Habilitar o rastreamento ativo em um tópico do Amazon SNS \(AWS SDK\)](#)
- [Habilitar o rastreamento ativo em um tópico do Amazon SNS \(AWS CLI\)](#)
- [Habilitar o rastreamento ativo em um tópico do Amazon SNS \(AWS CloudFormation\)](#)
- [Verificar se o rastreamento ativo está habilitado para o tópico](#)
- [Testar rastreamento ativo](#)

## Permissões de rastreamento ativo

Ao usar o console do Amazon SNS, o Amazon SNS tenta criar as permissões necessárias para que o tópico do Amazon SNS chame o X-Ray. A tentativa poderá ser rejeitada se você não tiver permissões suficientes para usar o console do Amazon SNS. Para obter mais informações, consulte [Casos de exemplo para controle de acesso do Amazon SNS](#) e [Gerenciamento de identidade e acesso no Amazon SNS](#).

Ao usar a CLI, você deve configurar manualmente as permissões. Essas permissões são configuradas usando políticas de recursos. Para saber mais sobre como usar as permissões necessárias no X-Ray, consulte [Amazon SNS e AWS X-Ray](#).

## Habilitar o rastreamento ativo em um tópico do Amazon SNS (console)

Quando o rastreamento ativo é habilitado em um tópico do Amazon SNS, ele lê o ID de rastreamento, envia os dados para o cliente com base no ID de rastreamento e propaga o ID de rastreamento para serviços downstream.

1. Faça login no console [do Amazon SNS](#).
2. Selecione um tópico ou crie um. Para obter mais detalhes sobre como criar tópicos, consulte [Criar um tópico do Amazon SNS](#).
3. Na página Criar tópico, na seção Detalhes, selecione um tipo de tópico: FIFO ou Padrão.
  - a. Insira um Name (Nome) para o tópico.
  - b. (Opcional) Insira um Display name (Nome para exibição) para o tópico.
4. Expanda Active tracing (Rastreamento ativo) e escolha Use active tracing (Usar rastreamento ativo).

Depois de habilitar o X-Ray para seu tópico do Amazon SNS, você pode usar o [mapa do serviço X-Ray](#) para visualizar os end-to-end rastreamentos e mapas de serviço do tópico.

## Habilitar o rastreamento ativo em um tópico do Amazon SNS (AWS SDK)

O exemplo de código a seguir mostra como habilitar o rastreamento ativo em um tópico do Amazon SNS usando o AWS SDK para Java.

```
public static void enableActiveTracing(SnsClient snsClient, String topicArn) {  
  
    try {  
  
        SetTopicAttributesRequest request = SetTopicAttributesRequest.builder()  
            .attributeName("TracingConfig")  
            .attributeValue("Active")  
            .topicArn(topicArn)  
            .build();  
  
        SetTopicAttributesResponse result = snsClient.setTopicAttributes(request);  
        System.out.println("\n\nStatus was " +  
result.sdkHttpResponse().statusCode() + "\n\nTopic " + request.topicArn()  
            + " updated " + request.attributeName() + " to " +  
request.attributeValue());  
  
    } catch (SnsException e) {  
        System.err.println(e.awsErrorDetails().errorMessage());  
    }  
}
```

## Habilitar o rastreamento ativo em um tópico do Amazon SNS (AWS CLI)

O exemplo de código a seguir mostra como habilitar o rastreamento ativo em um tópico do Amazon SNS usando a AWS CLI.

```
aws sns set-topic-attributes \  
  --topic-arn arn:aws:sns:us-west-2:123456789012:MyTopic \  
  --attribute-name TracingConfig \  
  --attribute-value Active
```

## Habilitar o rastreamento ativo em um tópico do Amazon SNS (AWS CloudFormation)

A pilha do AWS CloudFormation a seguir mostra como habilitar o rastreamento ativo em um tópico do Amazon SNS.

```
AWSTemplateFormatVersion: 2010-09-09  
Resources:  
  MyTopicResource:  
    Type: 'AWS::SNS::Topic'  
    Properties:  
      TopicName: 'MyTopic'  
      TracingConfig: 'Active'
```

## Verificar se o rastreamento ativo está habilitado para o tópico

Você pode usar o console do Amazon SNS para verificar se o rastreamento ativo está habilitado para o tópico ou para verificar quando a política de recursos não foi adicionada.

1. Faça login no console [do Amazon SNS](#).
2. No painel de navegação à esquerda, selecione Tópicos.
3. Na página Topics (Tópicos), escolha um tópico.
4. Escolha a guia Integrations (Integrações).

Quando o rastreamento ativo está habilitado, um ícone Active (Ativo) é exibido.

5. Se você habilitou o rastreamento ativo e não vê que a política de recursos foi adicionada, escolha Create policy (Criar política) para adicionar as outras permissões necessárias.

[Amazon SNS](#) > [Topics](#) > SampleTopic

## SampleTopic

Edit

Delete

Publish message

### Details

Name	Display name
SampleTopic	-
ARN	Topic owner
arn:aws:sns:us-east-1:242420583777:DeliveryRequest	123456789123
Type	
Standard	

< [Resource policy](#)[Delivery retry policy \(HTTP/S\)](#)[Delivery status logging](#)[Encryption](#)**[Integrations](#)**

&gt;

### AWS X-Ray active tracing

**Active tracing may require additional permission.**

We couldn't find an AWS X-Ray resource policy that allows Amazon SNS to send trace data. To create that policy now, choose "Create policy".

[Create policy](#)

Active tracing

 Active

Resource policy

 Not found

## Testar rastreamento ativo

1. Faça login no console [do Amazon SNS](#).
2. Crie um tópico do Amazon SNS. Para obter detalhes sobre como fazer isso, consulte [Para criar um tópico usando o AWS Management Console](#).
3. Expanda Active tracing (Rastreamento ativo) e escolha Use active tracing (Usar rastreamento ativo).
4. Publique uma mensagem no tópico do Amazon SNS. Para obter detalhes sobre como fazer isso, consulte [Para publicar mensagens nos tópicos do Amazon SNS usando o AWS Management Console](#).
5. Use o [mapa do serviço X-Ray](#) para visualizar os end-to-end rastreamentos e os mapas de serviço do tópico.



## Histórico de documentação

A tabela a seguir descreve alterações recentes no Guia do desenvolvedor do Amazon Simple Notification Service.

Às vezes, os recursos do serviço são lançados de forma incremental nas AWS regiões em que um serviço está disponível. Atualizamos esta documentação apenas para a primeira versão. Não fornecemos informações sobre a disponibilidade da região nem anunciamos lançamentos subsequentes da região. Para obter informações sobre a disponibilidade de recursos de serviço na região e para assinar notificações sobre atualizações, consulte [O que há de novo em AWS?](#) .

Alteração	Descrição	Data
<a href="#">Suporte do Oeste do Canadá (Calgary) para tópicos do FIFO</a>	O Amazon SNS oferece suporte ao tópico FIFO no Oeste do Canadá (Calgary).	28 de março de 2024
<a href="#">Suporte por SMS do Amazon SNS em cinco novas regiões</a>	O Amazon SNS adicionou suporte por SMS às seguintes regiões: Ásia-Pacífico (Hyderabad), Ásia-Pacífico (Melbourne), Oriente Médio (Emirados Árabes Unidos), Europa (Zurique) e Europa (Espanha).	8 de fevereiro de 2024
<a href="#">Suporte para HTTP v1 do Firebase Cloud Messaging (FCM)</a>	O Amazon SNS oferece suporte às credenciais do FCM v1.	18 de janeiro de 2024
<a href="#">Amazon SNS oferece suporte a SMS na região Ásia-Pacífico (Jacarta)</a>	O Amazon SNS oferece suporte a mensagens SMS na região Ásia-Pacífico (Jacarta).	14 de dezembro de 2023
<a href="#">AWS CloudFormation suporte para configuração de tópicos do DeliveryStatusLogging Amazon SNS</a>	AWS CloudFormation o suporte está disponível para configuração DeliveryStatusLogging durante	7 de dezembro de 2023

---

	a criação ou atualização de tópicos do Amazon SNS.	
<a href="#">Novos operadores de filtragem de mensagens adicionados</a>	Agora você pode usar os operadores de correspondência de sufixos, igual-ignorar-capitalização e OR ao filtrar mensagens do Amazon SNS.	16 de novembro de 2023
<a href="#">Suporte adicionado para arquivamento e reprodução de mensagens.</a>	Os proprietários de tópicos podem arquivar mensagens em um tópico por até 365 dias. Os assinantes do tópico podem reproduzir as mensagens arquivadas em um endpoint inscrito com o objetivo de recuperar mensagens em virtude de uma falha em uma aplicação subsequente ou para replicar o estado de uma aplicação existente.	26 de outubro de 2023
<a href="#">Suporte adicionado para inscrever uma fila padrão em um tópico FIFO</a>	Você pode inscrever uma fila FIFO ou uma fila padrão do Amazon SQS em um tópico FIFO do Amazon SNS. Somente as filas FIFO do Amazon SQS garantem que as mensagens sejam recebidas em ordem e sem duplicatas.	14 de setembro de 2023
Suporte de SMS adicionado para Israel (Tel Aviv)	O Amazon SNS SMS agora é compatível na região de Israel (Tel Aviv).	28 de agosto de 2023

---

<a href="#">Suporte ao rastreamento ativo do X-Ray adicionado a tópicos FIFO</a>	Anteriormente suportado apenas com tópicos padrão do Amazon SNS, AWS X-Ray agora rastreia e analisa solicitações de usuários à medida que elas percorrem seus tópicos de FIFO até suas assinaturas de endpoint Amazon Data Firehose, AWS Lambda Amazon SQS e HTTP/S.	31 de maio de 2023
<a href="#">Suporte avançado para o cabeçalho Content-Type</a>	É possível definir o cabeçalho Content-Type na política de solicitação para especificar o tipo de mídia da notificação.	23 de março de 2023
<a href="#">Adição de suporte ao rastreamento ativo</a>	AWS X-Ray rastreia e analisa as solicitações dos usuários à medida que elas percorrem seus tópicos padrão do Amazon SNS até suas assinaturas de endpoint do Amazon Data Firehose, AWS Lambda Amazon SQS e HTTP/S.	8 de fevereiro de 2023
<a href="#">Registro de ID de remetente de Singapura</a>	Instruções adicionadas para registrar IDs de remetente em Singapura.	10 de janeiro de 2023
<a href="#">Filtragem de mensagens baseada em carga útil</a>	A filtragem baseada em carga útil permite filtrar mensagens com base na carga útil da mensagem e evitar os custos associados ao processamento de dados indesejados.	22 de novembro de 2022

---

<a href="#"><u>Adição do algoritmo de hash SHA256 para assinatura de mensagens do Amazon SNS</u></a>	Adição de compatibilidade com o algoritmo de hash SHA256 ao usar a assinatura de mensagens do Amazon SNS.	15 de setembro de 2022
<a href="#"><u>Adição de mais regiões às mensagens SMS</u></a>	O Amazon SNS oferece suporte a mensagens SMS nas seguintes regiões: África (Cidade do Cabo), Ásia-Pacífico (Osaka), Europa (Milão) e AWS GovCloud (Leste dos EUA).	9 de setembro de 2022
<a href="#"><u>Adição de compatibilidade com a proteção de dados de mensagens</u></a>	A proteção de dados de mensagens protege os dados publicados em seus tópicos do Amazon SNS usando políticas de proteção de dados para auditar e bloquear as informações confidenciais que se movem entre aplicativos AWS ou serviços.	8 de setembro de 2022
<a href="#"><u>Novo processo de registro de números de chamada gratuita</u></a>	Suporte adicionado para envio de mensagens do Amazon SNS usando números de telefone de chamada gratuita (TFN) para destinatários dos Estados Unidos.	1º de agosto de 2022

[Suporte para controle de acesso baseado em atributos \(ABAC\)](#)

Suporte adicionado para controle de acesso baseado em atributos (ABAC) para ações de API, incluindo Publish e PublishBatch . O ABAC é uma estratégia de autorização que define permissões de acesso com base em tags que podem ser anexadas a recursos do IAM, como usuários e funções do IAM, e a AWS recursos, como tópicos do Amazon SNS, para simplificar o gerenciamento de permissões.

10 de janeiro de 2022

[Suporte para autenticação baseada em token da Apple para notificações push](#)

É possível autorizar o Amazon SNS a enviar notificações por push para sua aplicação iOS ou macOS fornecendo informações que identifiquem você como desenvolvedor da aplicação.

28 de outubro de 2021

[Novos remetentes de mensagens SMS são colocados na sandbox de SMS](#)

A sandbox de SMS existe para ajudar a evitar fraudes e abusos e para ajudar a proteger sua reputação como remetente. Enquanto sua AWS conta estiver na sandbox de SMS, você poderá enviar mensagens SMS somente para números de telefone de destino verificados.

1º de junho de 2021

[Novos remetentes de mensagens SMS são colocados na sandbox de SMS](#)

A sandbox de SMS existe para ajudar a evitar fraudes e abusos e para ajudar a proteger sua reputação como remetente. Enquanto sua AWS conta estiver na sandbox de SMS, você poderá enviar mensagens SMS somente para números de telefone de destino verificados.

1º de junho de 2021

[Novos atributos para enviar mensagens SMS para destinatários na Índia](#)

Dois novos atributos, ID da entidade e ID de modelo Agora são necessários para enviar mensagens SMS para destinatários na Índia.

22 de abril de 2021

[Atualizações para os operadores de filtragem](#)

Um novo operador, `cidr`, está disponível para correspondência de endereços IP de origem de mensagem e sub-redes. Agora você também pode verificar a ausência de uma chave de atributo, e usar um prefixo com `anything-but` operador para correspondência de string de atributo.

7 de abril de 2021

[Finalizando o suporte para códigos longos P2P para destinos dos EUA](#)

A partir de 1º de junho de 2021, os provedores de telecomunicações dos EUA não oferecem mais suporte ao uso de códigos longos person-to-person (P2P) para comunicações application-to-person (A2P) com destinos nos EUA. Em vez disso, você pode usar códigos curtos, números de ligação gratuita ou um novo tipo de número de origem chamado 10DLC.

16 de fevereiro de 2021

[Support para métricas de 1 minuto da Amazon CloudWatch](#)

A CloudWatch métrica de 1 minuto para o Amazon SNS agora está disponível em AWS todas as regiões.

28 de janeiro de 2021

[Support para endpoints Amazon Data Firehose](#)

Você pode inscrever streams de entrega do Firehose em tópicos do SNS. Isso permite que você envie notificações para endpoints de armazenamento e análise, como buckets do Amazon Simple Storage Service (Amazon S3), tabelas do Amazon Redshift, Amazon Service (Service) e muito mais. OpenSearch OpenSearch

12 de janeiro de 2021

[Os números de origem estão disponíveis](#)

Você pode usar números de origem ao enviar mensagens de texto (SMS).

23 de outubro de 2020

---

<a href="#">Suporte para tópicos FIFO do Amazon SNS</a>	Para integrar aplicativos distribuídos que exigem consistência de dados quase em tempo real, você pode usar tópicos FIFO (first in, first out) do Amazon SNS com filas FIFO do Amazon SQS.	22 de outubro de 2020
<a href="#">A biblioteca do cliente em versão ampliada do Amazon SNS para Java está disponível</a>	Você pode usar essa biblioteca para publicar mensagens grandes do Amazon SNS.	25 de agosto de 2020
<a href="#">O SSE está disponível nas regiões da China</a>	A criptografia no lado do servidor (SSE) para o Amazon SNS está disponível nas regiões da China.	20 de janeiro de 2020
<a href="#">Suporte para usar DLQs para capturar mensagens que não podem ser entregues</a>	É possível usar uma fila de mensagens mortas (DLQ) do Amazon SQS com uma assinatura do Amazon SNS para capturar mensagens que não podem ser entregues.	14 de novembro de 2019
<a href="#">Suporte para especificar valores de cabeçalho APNs personalizados</a>	Você pode especificar um valor personalizado do cabeçalho do APNs	18 de outubro de 2019
<a href="#">Suporte para a o campo de cabeçalho apns-push-type para APNs</a>	É possível usar o campo de cabeçalho apns-push-type para notificações móveis enviadas por meio do APNs.	10 de setembro de 2019
<a href="#">Support para solução de problemas de tópicos usando AWS X-Ray</a>	É possível usar o X-Ray para solucionar problemas com mensagens que passam pelos tópicos do SNS.	24 de julho de 2019

[Suporte para correspondência de chave de atributo usando o operador exists](#)

É possível usar o operador `exists` para verificar se uma mensagem recebida tem um atributo cuja chave está listada na política de filtro.

5 de julho de 2019

[Suporte para correspondência “tudo, exceto” de vários valores numéricos](#)

Além de várias strings, o Amazon SNS permite a correspondência “tudo, exceto” de vários valores numéricos.

5 de julho de 2019

[As notas de release do Amazon SNS estão disponíveis como um feed RSS](#)

Seguindo o título desta página (Histórico de documentação), escolhaRSS.

22 de junho de 2019

# Glossário do AWS

Para obter a terminologia mais recente da AWS, consulte o [glossário da AWS](#) na Referência do Glossário da AWS.

As traduções são geradas por tradução automática. Em caso de conflito entre o conteúdo da tradução e da versão original em inglês, a versão em inglês prevalecerá.